PYTHON CODING CHALLENGE – CAREERHUB

Problem Statement: A Job Board scenario is a digital platform or system that facilitates the process of job searching and recruitment. In this scenario, various stakeholders, such as job seekers, companies, and recruiters, use the platform to post, search for, and apply to job opportunities.

Create SQL Schema from the application, use the class attributes for table column names

```
Create Database careerhub;
Use careerhub;
```

CREATING TABLES :

JobListing Class:

```
create table JobListing (
    JobID INT primary key,
    CompanyID INT,
    JobTitle VARCHAR(100),
    JobDescription TEXT,
    JobLocation VARCHAR(100),
    Salary DECIMAL(10, 2) check (Salary >= 0),
    JobType VARCHAR(50),
    PostedDate DATETIME,
    foreign key (CompanyID) references Company(CompanyID)
);
```

Company Class:

```
create table Company (
    CompanyID INT primary key,
    CompanyName VARCHAR(100),
    Location VARCHAR(100)
);
```

Applicant Class:

```
create table Applicant (
    ApplicantID INT primary key,
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    Email VARCHAR(100),
    Phone VARCHAR(20),
    Resume VARCHAR(255)
);
```

JobApplication Class :

```
create table JobApplication (
   ApplicationID INT primary key,
   JobID INT,
   ApplicantID INT,
   ApplicationDate DATETIME,
   CoverLetter TEXT,
   foreign key (JobID) references JobListing(JobID),
   foreign key (ApplicantID) references Applicant(ApplicantID)
);
```

INSERTING VALUES :

JobListing Table

```
INSERT INTO JobListing (JobID, CompanyID, JobTitle, JobDescription, JobLocation, Salary,
JobType, PostedDate) VALUES
(101, 1, 'Software Engineer', 'Develop backend services', 'Chennai', 55000.00, 'Full-Time', NOW()),
(102, 2, 'DevOps Engineer', 'Handle CI/CD pipelines', 'Bangalore', 60000.00, 'Full-Time', NOW()),
(103, 3, 'AI Researcher', 'Research deep learning', 'Hyderabad', 75000.00, 'Full-Time', NOW()),
(104, 4, 'Data Analyst', 'Analyze business data', 'Pune', 48000.00, 'Contract', NOW()),
(105, 5, 'Cybersecurity Specialist', 'Monitor threats', 'Delhi', 65000.00, 'Full-Time', NOW());
```

Company Table:

```
INSERT INTO Company (CompanyID, CompanyName, Location) VALUES
(1, 'Tech Innovators', 'Chennai'),
(2, 'Cloudify Corp', 'Bangalore'),
(3, 'AI Works', 'Hyderabad'),
(4, 'DataSpring', 'Pune'),
(5, 'SecureNet', 'Delhi');
```

Applicant Table :

```
INSERT INTO Applicant (ApplicantID, FirstName, LastName, Email, Phone, Resume)
VALUES
(201, 'Arun', 'Kumar', 'arun.kumar@example.com', '9876543210', 'arun_resume.pdf'),
(202, 'Meena', 'Ravi', 'meena.ravi@domain.com', '8765432109', 'meena_resume.docx'),
(203, 'Sundar', 'Raj', 'sundar.raj@mail.com', '7654321098', 'sundar_resume.txt'),
(204, 'Lakshmi', 'Narayanan', 'lakshmi.n@techworld.com', '9988776655',
'lakshmi_resume.pdf'),
(205, 'Vignesh', 'Balaji', 'vignesh.balaji@cloud.com', '8899776655', 'vignesh_resume.docx');
```

JobApplication Table

INSERT INTO JobApplication (ApplicationID, JobID, ApplicantID, ApplicationDate, CoverLetter) VALUES
(301, 101, 201, NOW(), 'Excited to work with backend systems.'),
(302, 102, 202, NOW(), 'DevOps is my strength and passion.'),
(303, 103, 203, NOW(), 'Deep learning and AI are my focus.'),
(304, 104, 204, NOW(), 'Experienced in business data analysis.'),
(305, 105, 205, NOW(), 'I love securing systems and networks.');

1.Create and implement the mentioned class and the structure in your application.

JobListing Class:

```python
class JobListing:
    def __init__(self, job_id, company_id, job_title, job_description, job_location, salary, job_type, posted_date):
        self.job_id = job_id
        self.company_id = company_id
        self.job_title = job_title
        self.job_description = job_description
        self.job_location = job_location
        self.salary = salary
        self.job_type = job_type
        self.posted_date = posted_date
        self.applicants = []

    def apply(self, applicant_id: int, cover_letter: str):
        self.applicants.append((applicant_id, cover_letter))

    def get_applicants(self):
        return self.applicants
```

Company Class:

```python
class Company:
    def __init__(self, company_id, company_name, location):
        self.company_id = company_id
        self.company_name = company_name
        self.location = location
        self.jobs = []

    def post_job(self, job_id, job_title, job_description, job_location, salary, job_type):
        posted_date = datetime.now()
        job = JobListing(job_id, self.company_id, job_title, job_description, job_location, salary, job_type, posted_date)
        self.jobs.append(job)
        return job

    def get_jobs(self):
        return self.jobs
```

Applicant Class:

```
class Applicant:
    def __init__(self, applicant_id, first_name, last_name, email, phone, resume):
        self.applicant_id = applicant_id
        self.first_name = first_name
        self.last_name = last_name
        self.email = email
        self.phone = phone
        self.resume = resume
        self.applied_jobs = []

    def create_profile(self, email, first_name, last_name, phone):
        self.email = email
        self.first_name = first_name
        self.last_name = last_name
        self.phone = phone

    def apply_for_job(self, job_id, cover_letter):
        application_date = datetime.now()
        application = JobApplication(None, job_id, self.applicant_id, application_date, cover_letter)
        self.applied_jobs.append(application)
        return application
```

JobApplication Class :

```
class JobApplication:
    def __init__(self, application_id, job_id, applicant_id, application_date, cover_letter):
        self.application_id = application_id
        self.job_id = job_id
        self.applicant_id = applicant_id
        self.application_date = application_date
        self.cover_letter = cover_letter
```

2.DatabaseManager Class:

DAO/Interface class :

Methods:

• InitializeDatabase(): Initializes the database schema and tables.

• InsertJobListing(job: JobListing): Inserts a new job listing into the "Jobs" table.

• InsertCompany(company: Company): Inserts a new company into the "Companies" table.

• InsertApplicant(applicant: Applicant): Inserts a new applicant into the "Applicants" table.

• InsertJobApplication(application: JobApplication): Inserts a new job application into the "Applications" table.

• GetJobListings(): List: Retrieves a list of all job listings. • GetCompanies(): List: Retrieves a list of all companies.

• GetApplicants(): List: Retrieves a list of all applicants.

• GetApplicationsForJob(jobID: int): List: Retrieves a list of job applications for a specific job listing.

DatabaseManagerInterface.py > ...

```python
from abc import ABC, abstractmethod

class DatabaseManagerInterface(ABC):

    @abstractmethod
    def initialize_database(self): pass

    @abstractmethod
    def insert_job_listing(self, job): pass

    @abstractmethod
    def insert_company(self, company): pass

    @abstractmethod
    def insert_applicant(self, applicant): pass

    @abstractmethod
    def insert_job_application(self, application): pass

    @abstractmethod
    def get_job_listings(self): pass

    @abstractmethod
    def get_companies(self): pass

    @abstractmethod
    def get_applicants(self): pass

    @abstractmethod
    def get_applications_for_job(self, job_id): pass
```

DAO/Implementation class :

3.Exceptions handling :

Create and implement the following exceptions in your application.

• Invalid Email Format Handling:

• Salary Calculation Handling:

• File Upload Exception Handling:

• Application Deadline Handling:

• Database Connection Handling:

```python
class InvalidEmailException(Exception): pass

class NegativeSalaryException(Exception): pass

class FileUploadException(Exception): pass

class ApplicationDeadlineException(Exception): pass

class DBConnectionException(Exception): pass
```

4.Database Connectivity :

Create and implement the following tasks in your application

DBConnUtil

```python
import mysql.connector
from util.DBPropertyUtil import get_property
from exception.CustomExceptions import DBConnectionException

class DBConnUtil:
    @staticmethod
    def get_connection():
        try:
            props = get_property("config/db.properties")
            return mysql.connector.connect(
                host=props["DB_HOST"],
                user=props["DB_USER"],
                password=props["DB_PASSWORD"],
                database=props["DB_NAME"]
            )
        except Exception as e:
            raise DBConnectionException(f"Database connection failed: {e}")
```

DBPropertyUtil

DBPropertyUtil.py > get_property

```python
def get_property(filename):
    props = {}
    with open(filename, 'r') as file:
        for line in file:
            if '=' in line:
                key, val = line.strip().split('=', 1)
                props[key.strip()] = val.strip()
    return props
```

Db.Properties

config > db.properties

```
DB_HOST = localhost
DB_USER = root
DB_PASSWORD = dharshini2004
DB_NAME = careerhub
```

Main Class

MainModule.py > ...

```python
from dao.DatabaseManagerImpl import DatabaseManagerImpl
from entity.Company import Company
from entity.Applicant import Applicant
from entity.JobListing import JobListing
from entity.JobApplication import JobApplication
from exception.CustomExceptions import InvalidEmailException
from exception.CustomExceptions import NegativeSalaryException
from exception.CustomExceptions import FileUploadException
from exception.CustomExceptions import  ApplicationDeadlineException
from exception.CustomExceptions import DBConnectionException
from datetime import datetime
import re
import os
def validate_email(email):
    if not re.match(r"[^@]+@[^@]+\.[^@]+", email):
        raise InvalidEmailException("❌ Invalid email format.")

def calculate_average_salary(job_listings):
    total_salary = 0
    count = 0
    for job in job_listings:
        salary = job[5]
        if salary < 0:
            raise NegativeSalaryException(f"❌ Negative salary found in job ID: {job[0]}")
        total_salary += salary
        count += 1
    return total_salary / count if count > 0 else 0

def upload_resume(file_path):
    if not os.path.exists(file_path):
        raise FileUploadException("❌ File not found.")
    if not file_path.endswith(('.pdf', '.docx', '.txt')):
        raise FileUploadException("❌ Unsupported file format. Use PDF, DOCX, or TXT.")
    if os.path.getsize(file_path) > 2 * 1024 * 1024:
        raise FileUploadException("❌ File size exceeds 2MB.")
    return file_path
```

```python
def check_application_deadline(deadline):
    if datetime.now() > deadline:
        raise ApplicationDeadlineException("❌ Application deadline has passed.")

def main():
    try:
        db = DatabaseManagerImpl()
    except DBConnectionException as e:
        print(e)
        return

    while True:
        print("\n--- CareerHub Menu ---")
        print("1. Add Company")
        print("2. Add Job Listing")
        print("3. Add Applicant")
        print("4. Apply for Job")
        print("5. View Job Listings")
        print("6. View Applicants")
        print("7. View Applications for a Job")
        print("8. Calculate Average Salary (with validation)")
        print("9. Upload Resume File (with validation)")
        print("10. Check Application Deadline")
        print("0. Exit")

        choice = input("Enter your choice: ")

        try:
            if choice == "1":
                cid = int(input("Company ID: "))
                name = input("Company Name: ")
                loc = input("Location: ")
                db.insert_company(Company(cid, name, loc))
                print("✅ Company added.")
```

```python
    elif choice == "2":
        jid = int(input("Job ID: "))
        cid = int(input("Company ID: "))
        title = input("Title: ")
        desc = input("Description: ")
        loc = input("Location: ")
        salary = float(input("Salary: "))
        if salary < 0:
            raise NegativeSalaryException("❌ Salary cannot be negative.")
        jtype = input("Job Type: ")
        date = datetime.now()
        db.insert_job_listing(JobListing(jid, cid, title, desc, loc, salary, jtype, date))
        print("✅ Job listing added.")

    elif choice == "3":
        aid = int(input("Applicant ID: "))
        fname = input("First Name: ")
        lname = input("Last Name: ")
        email = input("Email: ")
        validate_email(email)
        phone = input("Phone: ")
        resume = input("Resume file path: ")
        resume_path = upload_resume(resume)
        db.insert_applicant(Applicant(aid, fname, lname, email, phone, resume_path))
        print("✅ Applicant registered.")

    elif choice == "4":
        appid = int(input("Application ID: "))
        jid = int(input("Job ID: "))
        aid = int(input("Applicant ID: "))
        deadline_str = input("Enter application deadline (YYYY-MM-DD HH:MM): ")
        deadline = datetime.strptime(deadline_str, "%Y-%m-%d %H:%M")
        check_application_deadline(deadline)
        cover = input("Cover Letter: ")
        date = datetime.now()
        db.insert_job_application(JobApplication(appid, jid, aid, date, cover))
```

```
        elif choice == "5":
            for job in db.get_job_listings():
                print(job)

        elif choice == "6":
            for applicant in db.get_applicants():
                print(applicant)

        elif choice == "7":
            jid = int(input("Job ID: "))
            for app in db.get_applications_for_job(jid):
                print(app)

        elif choice == "8":
            jobs = db.get_job_listings()
            avg = calculate_average_salary(jobs)
            print(f"✅ Average salary offered: {avg:.2f}")

        elif choice == "9":
            path = input("Enter resume file path: ")
            result = upload_resume(path)
            if result:
                print("✅ Resume upload validated.")

        elif choice == "10":
            deadline_str = input("Enter application deadline (YYYY-MM-DD HH:MM): ")
            deadline = datetime.strptime(deadline_str, "%Y-%m-%d %H:%M")
            check_application_deadline(deadline)
            print("✅ You can still apply for the job.")

        elif choice == "0":
            print("👋 Exiting CareerHub.")
            break

        else:
```

```
            print("✅ You can still apply for the job.")

        elif choice == "0":
            print("👋 Exiting CareerHub.")
            break

        else:
            print("❌ Invalid option.")

    except (InvalidEmailException, NegativeSalaryException, FileUploadException, ApplicationDeadlineException) as e:
        print(e)
    except ValueError:
        print("❌ Invalid input format. Please enter correct values.")
    except Exception as e:
        print(f"❌ An unexpected error occurred: {e}")

if __name__ == "__main__":
    main()
```

Output 1 :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\murug\Downloads\Careerhub> python -m main.MainModule

--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 1
Company ID: 6
Company Name: data hub
Location: chennai
✅ Company added.
```

Output 2 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 2
Job ID: 106
Company ID: 6
Title: Data Science
Description: Analysing Data
Location: Bangalore
Salary: 400000
Job Type: Part - Time
✅ Job listing added.
```

Output 3 :

```
--- CareerHub Menu ---
1. Add Company
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
3. Add Applicant
4. Apply for Job
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 3
Applicant ID: 206
First Name: Dharshini
Last Name: Murugaiyan
Email: dharhini@example.com
Phone: 8367328190
Resume file path: dharsh_resume.txt
❌ File not found.
```

Output 4 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 4
Application ID: 207
Job ID: 107
Applicant ID: 208
Enter application deadline (YYYY-MM-DD HH:MM): 2025-04-12 12:34
❌ Application deadline has passed.
```

Output 5 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 5
(101, 1, 'Software Engineer', 'Develop backend services', 'Chennai', Decimal('55000.00'), 'Full-Time', datetime.datetime(2025, 6, 27, 12, 38, 4
8))
(102, 2, 'DevOps Engineer', 'Handle CI/CD pipelines', 'Bangalore', Decimal('60000.00'), 'Full-Time', datetime.datetime(2025, 6, 27, 12, 38, 48)
)
(103, 3, 'AI Researcher', 'Research deep learning', 'Hyderabad', Decimal('75000.00'), 'Full-Time', datetime.datetime(2025, 6, 27, 12, 38, 48))
(104, 4, 'Data Analyst', 'Analyze business data', 'Pune', Decimal('48000.00'), 'Contract', datetime.datetime(2025, 6, 27, 12, 38, 48))
(105, 5, 'Cybersecurity Specialist', 'Monitor threats', 'Delhi', Decimal('65000.00'), 'Full-Time', datetime.datetime(2025, 6, 27, 12, 38, 48))
(106, 6, 'Data Science', 'Analysing Data', 'Bangalore', Decimal('400000.00'), 'Part - Time', datetime.datetime(2025, 6, 27, 13, 3, 27))
```

Output 6 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 6
(201, 'Arun', 'Kumar', 'arun.kumar@example.com', '9876543210', 'arun_resume.pdf')
(202, 'Meena', 'Ravi', 'meena.ravi@domain.com', '8765432109', 'meena_resume.docx')
(203, 'Sundar', 'Raj', 'sundar.raj@mail.com', '7654321098', 'sundar_resume.txt')
(204, 'Lakshmi', 'Narayanan', 'lakshmi.n@techworld.com', '9988776655', 'lakshmi_resume.pdf')
(205, 'Vignesh', 'Balaji', 'vignesh.balaji@cloud.com', '8899776655', 'vignesh_resume.docx')
```

Output 7 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 7
Job ID: 104
(304, 104, 204, datetime.datetime(2025, 6, 27, 12, 39, 11), 'Experienced in business data analysis.')
```

Output 8 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 8
✅ Average salary offered: 117166.67
```

Output 9 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 9
Enter resume file path: arun_resume.pdf
❌ File not found.
```

Output 10 :

```
--- CareerHub Menu ---
1. Add Company
2. Add Job Listing
3. Add Applicant
4. Apply for Job
5. View Job Listings
6. View Applicants
7. View Applications for a Job
8. Calculate Average Salary (with validation)
9. Upload Resume File (with validation)
10. Check Application Deadline
0. Exit
Enter your choice: 10
Enter application deadline (YYYY-MM-DD HH:MM): 2025-06-27  12:34
❌ Application deadline has passed.
```