**Coding Challenges: CareerHub, The Job Board-By(Akkala Jayanth)**

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

**Job Class:**

**JobID (int): A unique identifier for each job listing.**

**• CompanyID (int): A reference to the company offering the job.**

**• JobTitle (string): The title of the job.**

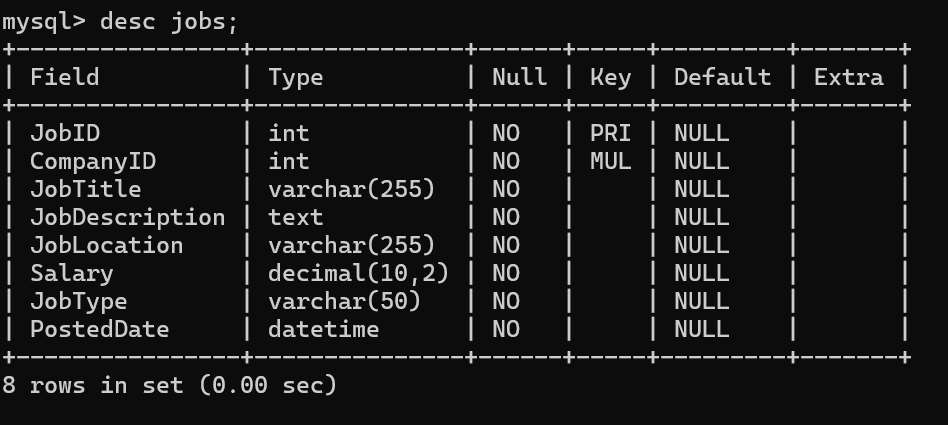
**• JobDescription (string): A detailed description of the job.**

**• JobLocation (string): The location of the job.**

**• Salary (decimal): The salary offered for the job.**

**• JobType (string): The type of job (e.g., Full-time, Part-time, Contract).**

**• PostedDate (DateTime): The date when the job was posted**

****

**Applicant Class:**

**ApplicantID (int): A unique identifier for each applicant.**

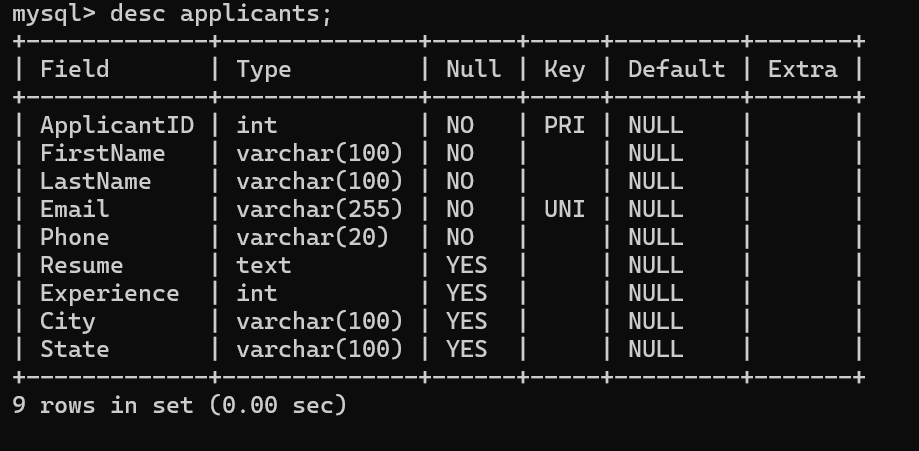
**• FirstName (string): The first name of the applicant.**

**• LastName (string): The last name of the applicant.**

**• Email (string): The email address of the applicant.**

**• Phone (string): The phone number of the applicant.**

**• Resume (string): The applicant's resume or a reference to the resume file.**

****

**JobApplication Class:**

**ApplicationID (int): A unique identifier for each job application.**

**• JobID (int): A reference to the job listing.**

**• ApplicantID (int): A reference to the applicant.**

**• ApplicationDate (DateTime): The date and time when the application was submitted.**

**• CoverLetter (string): The cover letter submitted with the application.**

**A screenshot of a computer program

Description automatically generated**

**2.DatabaseManager Class:**

**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<JobListing>: Retrieves a list of all job listings.**

**• GetCompanies(): List<Company>: Retrieves a list of all companies.**

**• GetApplicants(): List<Applicant>: Retrieves a list of all applicants.**

**• GetApplicationsForJob(jobID: int): List<JobApplication>: Retrieves a list of job applications for a**

**specific job listing.**

* There are several factors that has to be consider the creation of the database properties such as the util package and the connectivity package has to be maintained in a proper structure such as :

A screen shot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated

**Exceptions handling** Create and implement the following exceptions in your application.

• **Invalid Email Format Handling**: o In the Job Board application, during the applicant registration process, users are required to enter their email addresses. Write a program that prompts the user to input an email address and implement exception handling to ensure that the email address follows a valid format (e.g., contains "@" and a valid domain). If the input is not valid, catch the exception and display an error message. If it is valid, proceed with registration.

* While writing a exception you have to get know about the try and catch block based on the exceptions that your are writing .
* For example, while you are writing the email exception then u have to mention that u cant write an email without an “@” symbol

A screen shot of a computer program

Description automatically generated

• **Salary Calculation Handling**: o Create a program that calculates the average salary offered by companies for job listings. Implement exception handling to ensure that the salary values are non-negative when computing the average. If any salary is negative or invalid, catch the exception and display an error message, indicating the problematic job listings.

* While writing an exception for salary u have make sure the sum of salary should be greater then the value 0 such that there is no negative value will be calculated at the end .  
  A screen shot of a computer code

  Description automatically generated

**• File Upload Exception Handling**: o In the Job Board application, applicants can upload their resumes as files. Write a program that handles file uploads and implements exception handling to catch and handle potential errors, such as file not found, file size exceeded, or file format not supported. Provide appropriate error messages in each case.

* While writing the file upload exception u have to make sure that the application resumes must be uploaded in such a way that their should be no file that was not uploaded , and also the naming convensions should be in the order as firstname , last name , cv or resume.

A screen shot of a computer program

Description automatically generated

**Application Deadline Handling**: o Develop a program that checks whether a job application is submitted before the application deadline. Implement exception handling to catch situations where an applicant tries to submit an application after the deadline has passed. Display a message indicating that the application is no longer accepted.

* The application deadline is a kind of exception where the db should not accept the application after the deadline such that the application who applied before the deadline only should be accepted.

A screen shot of a computer

Description automatically generated

**Database Connection Handling**: o In the Job Board application, database connectivity is crucial. Create a program that establishes a connection to the database to retrieve job listings. Implement exception handling to catch database-related exceptions, such as connection errors or SQL query errors. Display appropriate error messages and ensure graceful handling of these exceptions.

* This is a type of exception where there is a error in the connection to the database such that the jvm throws this error . such that the user can get to know what kind of error they are getting such that they can work on it and resolve the issue .

A computer screen shot of text

Description automatically generated

**Database Connectivity**

Create and implement the following tasks in your application.

**• Job Listing Retrieval**: Write a program that connects to the database and retrieves all job listings from the "Jobs" table. Implement database connectivity using Entity Framework and display the job titles, company names, and salaries

* For getting the data from the joblisting u need to make a DAO(DATA ACCESS OBJECT) package such that u can create a connection with the database and get the data that are related for the job listings.

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

• **Applicant Profile Creation**: Create a program that allows applicants to create a profile by entering their information. Implement database connectivity to insert the applicant's data into the "Applicants" table. Handle potential database-related exceptions.

* Same as the jobs listing you to need to create the the dao files for the applicant such that u can make a creation for the applicant.

A screen shot of a computer screen

Description automatically generated

• **Job Application Submission**: Develop a program that allows applicants to apply for a specific job listing. Implement database connectivity to insert the job application details into the "Applications" table, including the applicant's ID and the job ID. Ensure that the program handles database connectivity and insertion exceptions

* To develop an application submission we need several functionalities to be considered such that an “Application deadline exception etc”.
* And we need several import functionalities such as sql jar files . which contains the several operations such as insert the query and update and etc .
* The optimised code for the application submission is :

A screen shot of a computer program

Description automatically generated

• **Company Job Posting**: Write a program that enables companies to post new job listings. Implement database connectivity to insert job listings into the "Jobs" table, including the company's ID. Handle database-related exceptions and ensure the job posting is successful.

* To write a program that posts a job has required several import functions as well as the database connectivity such that we have give the root id and password such that u have to make the necessary adjustments and make it possible to write a clean code with abstractions and interfaces

A screen shot of a computer program

Description automatically generated

**Salary Range Query**: Create a program that allows users to search for job listings within a specified salary range. Implement database connectivity to retrieve job listings that match the user's criteria, including job titles, company names, and salaries. Ensure the program handles database connectivity and query exceptions.

* To write the salary query we need to first connect to the db by giving the root access and also have to use several join to make the data accurate while inserting and data manupluations . and have to use several level of abstraction

A screen shot of a computer program

Description automatically generated

After completing all the functions and import functions you will get the output as follows  
A screenshot of a computer

Description automatically generated