## **Documentation- Employee Database**

### **Overview**

This Java application processes employee data by utilizing the Employee, EmployeeData, and Main classes. It demonstrates the use of the Function interface for data transformation, calculates the average salary, and performs filtering operations using Java Streams.

### **Classes and Methods**

#### **Employee Class**

**Description:** Represents an employee with attributes such as name, age, department, salary, and position. Provides getter and setter methods for these attributes.

**Attributes:**

* name (String): Employee's name.
* age (int): Employee's age.
* department (String): Department where the employee works.
* salary (double): Employee's salary.
* position (String): Employee's job position.

**Constructor:**

public Employee(String name, int age, String department, double salary, String position)

Initializes an Employee object with the specified attributes.

**Getters and Setters:**

* public String getName()
* public int getAge()
* public String getDepartment()
* public double getSalary()
* public String getPosition()
* public void setPosition(String position)

**Methods:**

* @Override public String toString()
  + Returns a string representation of the employee in the format: "name (position)".

#### **EmployeeData Class**

**Description:** Contains methods for retrieving employee data and defining functions for data transformation.

**Methods:**

* public static List<Employee> getEmployees()
  + Returns a list of sample Employee objects with predefined attributes.
* public static Function<Employee, String> getNameAndDepartment
  + A function that takes an Employee object and returns a string combining the employee's name and department.
* public static Function<Employee, String> getNameAndJobTitle
  + A function that takes an Employee object and returns a string combining the employee's name and job title.

#### **Main Class**

**Description:** The main class where the application runs. It demonstrates various operations using the Employee data, including transformation, filtering, and aggregation.

**Methods:**

* **Operations:**
  + **List of Name and Department:** Transforms employees to name:department and prints.
  + **List of Name and Job Title:** Transforms employees to name:position and prints.
  + **Average Salary:** Calculates and prints the average salary of employees.
  + **Filter by Age:** Filters and prints employees older than 25.
  + **Filter by Department:** Filters and prints employees in the IT department.

### **How to Run the Program**

1. **Compile the Code:**
   * Save each class (Employee.java, EmployeeData.java, and Main.java) in the same directory.
   * Open a terminal or command prompt.
   * Navigate to the directory where the files are saved.
   * Compile the code using the following command:

Employee.java EmployeeData.java Main.java

1. **Run the Program:**
   * After successful compilation, run the Main class using the following command:

java Main

1. **Expected Output:**
   * The program will print the lists of employee names with departments and job titles, average salary, and filtered results based on age and department to the console.