**Employee:**

public class Employee{  
 private String firstName;  
 private String secondName;  
 private String gender;  
 private int age;  
  
 public Employee(){}  
 public Employee(String firstName, String secondName, String gender, int age)  
 {  
 this.firstName = firstName;  
 this.secondName = secondName;  
 this.gender = gender;  
 this.age = age;  
 }  
  
 public String getFirstName() {  
 return firstName;  
 }  
  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
  
 public String getSecondName() {  
 return secondName;  
 }  
  
 public void setSecondName(String secondName) {  
 this.secondName = secondName;  
 }  
  
 public String getGender() {  
 return gender;  
 }  
  
 public void setGender(String gender) {  
 this.gender = gender;  
 }  
  
 public int getAge() {  
 return age;  
 }  
  
 public void setAge(int age) {  
 this.age = age;  
 }  
  
 public String toString() {  
 return "Employee{" +  
 "firstName='" + firstName + '\'' +  
 ", secondName='" + secondName + '\'' +  
 ", gender='" + gender + '\'' +  
 ", age=" + age +  
 '}';  
 }  
}

**CEO:**

public class CEO extends Employee{  
 private int salary;  
  
 public CEO()  
 {  
  
 }  
 public CEO(String firstName, String secondName, String gender, int age, int salary)  
 {  
 super(firstName, secondName, gender, age);  
 this.salary = salary;  
 }  
  
 public void setSalary(int salary) {  
 this.salary = salary;  
 }  
  
 public int getSalary() {  
 return salary;  
 }  
  
 @Override //own implementation of toString method for CEO. (Polymorphism)  
 public String toString() {  
 return "CEO{" + super.toString() +  
 " salary=" + salary +  
 '}';  
 }  
}

**Developer:**

public class CEO extends Employee{  
 private int salary;  
  
 public CEO()  
 {  
  
 }  
 public CEO(String firstName, String secondName, String gender, int age, int salary)  
 {  
 super(firstName, secondName, gender, age);  
 this.salary = salary;  
 }  
  
 public void setSalary(int salary) {  
 this.salary = salary;  
 }  
  
 public int getSalary() {  
 return salary;  
 }  
  
 @Override //own implementation of toString method for CEO. (Polymorphism)  
 public String toString() {  
 return "CEO{" + super.toString() +  
 " salary=" + salary +  
 '}';  
 }  
}

**Tester:**

public class CEO extends Employee{  
 private int salary;  
  
 public CEO()  
 {  
  
 }  
 public CEO(String firstName, String secondName, String gender, int age, int salary)  
 {  
 super(firstName, secondName, gender, age);  
 this.salary = salary;  
 }  
  
 public void setSalary(int salary) {  
 this.salary = salary;  
 }  
  
 public int getSalary() {  
 return salary;  
 }  
  
 @Override //own implementation of toString method for CEO. (Polymorphism)  
 public String toString() {  
 return "CEO{" + super.toString() +  
 " salary=" + salary +  
 '}';  
 }  
}

**Project:**

import org.w3c.dom.ls.LSOutput;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.PreparedStatement;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.InputMismatchException;  
import java.util.Scanner;  
  
public class Project {  
 private ArrayList<Employee> employees = new ArrayList<Employee>();  
 private String title;  
 private long directCost;  
 private long indirectCost;  
  
 public Project()  
 {}  
 public Project(ArrayList<Employee> employees, String title, long directCost, long indirectCost)  
 {  
 this.employees = employees;  
 this.title = title;  
 this.directCost = directCost;  
 this.indirectCost = indirectCost;  
 }  
  
 public ArrayList<Employee> getEmployees() {  
 return employees;  
 }  
  
 public void setEmployees(ArrayList<Employee> employees) {  
 this.employees = employees;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public void setTitle(String title) {  
 this.title = title;  
 }  
  
 public long getDirectCost() {  
 return directCost;  
 }  
  
 public void setDirectCost(long directCost) {  
 this.directCost = directCost;  
 }  
  
 public long getIndirectCost() {  
 return indirectCost;  
 }  
  
 public void setIndirectCost(long indirectCost) {  
 this.indirectCost = indirectCost;  
 }  
  
 public void addEmployee(Employee employee)  
 {  
 try{  
 employees.add(employee);  
 System.*out*.println("");  
 }  
 catch(NullPointerException exception)  
 {  
 exception.printStackTrace();  
 }  
 }  
  
 public boolean removeEmployee(String employeeName)  
 {  
 for(int i = 0; i < employees.size(); i++)  
 {  
 if(employees.get(i).getFirstName().equals(employeeName))  
 {  
 employees.remove(i);  
 System.*out*.println("You have kicked employee №" + (i+1) + " with name: " + employeeName);  
 return true;  
 }  
 else if(employees.get(i).getSecondName().equals(employeeName))  
 {  
 employees.remove(i);  
 System.*out*.println("You have kicked employee №" + (i+1) + " with name: " + employeeName);  
 return true;  
 }  
 }  
 return false;  
 }  
  
 public void printAllStaff()  
 {  
 System.*out*.println("All workers: ");  
 if(employees.size() != 0)  
 {  
 for(Employee employee : employees)  
 {  
 System.*out*.println(employee.getFirstName() + " " + employee.getSecondName());  
 }  
 }  
 else  
 {  
 System.*out*.println("No one works on project...");  
 }  
 }  
  
 public double printTotalCostProject()  
 {  
 System.*out*.print("Total cost of project: ");  
 return getDirectCost() + getIndirectCost();  
 }  
  
 public String toString() {  
 String employeesString = "";  
 for(Employee employee : employees)  
 {  
 employeesString += "\n" + employee;  
 }  
 return "Project{" +  
 "title='" + title + '\'' +  
 ", directCost=" + directCost +  
 ", indirectCost=" + indirectCost +  
 '}' + "\n" +" Employees in project: " + employeesString;  
 }  
  
}

**Database:**

import jdk.jshell.spi.SPIResolutionException;  
  
import javax.xml.transform.Result;  
import java.sql.\*;  
import java.util.ArrayList;  
import java.util.InputMismatchException;  
import java.util.Scanner;  
  
public class Database  
{  
 public static void insert()  
 {  
 Scanner in = new Scanner(System.*in*);  
 String URL = "jdbc:postgresql://localhost:5432/postgres";  
 String username = "postgres";  
 String password = "1245emer";  
  
 System.*out*.println("Insert employee or project (1 or 2)");  
 int choice = in.nextInt();  
 if(choice == 1)  
 {  
 try {  
 System.*out*.println("Select type of employee (CEO / Developer / Tester):");  
 String type = in.next();  
 if (type.equals("CEO"))  
 {  
 CEO ceo = new CEO();  
  
 System.*out*.println("Write first name of CEO: ");  
 String firstName = in.next();  
 ceo.setFirstName(firstName);  
  
 System.*out*.println("Write second name of CEO: ");  
 String secondName = in.next();  
 ceo.setSecondName(secondName);  
  
 System.*out*.println("Write gender of CEO");  
 String gender = in.next();  
 ceo.setGender(gender);  
  
 System.*out*.println("Write age of CEO");  
 int age = in.nextInt();  
 ceo.setAge(age);  
  
 System.*out*.println("Write salary of CEO");  
 int salary = in.nextInt();  
 ceo.setSalary(salary);  
  
 Connection connection = DriverManager.*getConnection*(URL, username, password);  
 String INSERT = "INSERT INTO ceo(project\_id, firstName, secondName, age, salary) VALUES(?, ?, ?, ?, ?)";  
 PreparedStatement statement = connection.prepareStatement(INSERT);  
 System.*out*.println("Select Project ID: ");  
 int id = in.nextInt();  
 statement.setInt(1, id);  
 statement.setString(2, ceo.getFirstName());  
 statement.setString(3, ceo.getSecondName());  
 statement.setInt(4, ceo.getAge());  
 statement.setInt(5, ceo.getSalary());  
  
 int executeUpdate = statement.executeUpdate();  
 }  
 if (type.equals("Developer")) {  
 Connection connection = DriverManager.*getConnection*(URL, username, password);  
 Developer developer1 = new Developer("Mark", "Andre", "male", 21, 20000, "Middle", "backend");  
 Developer developer = new Developer();  
  
 System.*out*.println("Write first name of Developer: ");  
 String firstName = in.next();  
 developer.setFirstName(firstName);  
  
 System.*out*.println("Write second name of Developer: ");  
 String secondName = in.next();  
 developer.setSecondName(secondName);  
  
 System.*out*.println("Write gender of Developer");  
 String gender = in.next();  
 developer.setGender(gender);  
  
 System.*out*.println("Write age of Developer");  
 int age = in.nextInt();  
 developer.setAge(age);  
  
 System.*out*.println("Write salary of Developer");  
 int salary = in.nextInt();  
 developer.setSalary(salary);  
  
 System.*out*.println("Write category of Developer(senior | middle | junior)");  
 String category = in.next();  
 developer.setCategory(category);  
  
 System.*out*.println("Write side of Developer(backend | frontend | full-stack)");  
 String side = in.next();  
 developer.setDeveloperSide(side);  
  
 String INSERT = "INSERT INTO developer(project\_id, salary, category, developerSide, firstname, secondname) VALUES(?, ?, ?, ?, ?, ?)";  
 PreparedStatement statement = connection.prepareStatement(INSERT);  
 System.*out*.println("Select Project ID: ");  
 int id = in.nextInt();  
 statement.setInt(1, id);  
 statement.setInt(2, developer.getSalary());  
 statement.setString(3, developer.getCategory());  
 statement.setString(4, developer.getDeveloperSide());  
 statement.setString(5, developer.getFirstName());  
 statement.setString(6, developer.getSecondName());  
  
 int executeUpdate = statement.executeUpdate();  
 }  
 if (type.equals("Tester")) {  
 Connection connection = DriverManager.*getConnection*(URL, username, password);  
 Tester tester1 = new Tester("Alisa", "Alexandrovna", "female", 22, 7800, "QA");  
 Tester tester = new Tester();  
  
 System.*out*.println("Write first name of Tester: ");  
 String firstName = in.next();  
 tester.setFirstName(firstName);  
  
 System.*out*.println("Write second name of Tester: ");  
 String secondName = in.next();  
 tester.setSecondName(secondName);  
  
 System.*out*.println("Write gender of Tester");  
 String gender = in.next();  
 tester.setGender(gender);  
  
 System.*out*.println("Write age of Tester");  
 int age = in.nextInt();  
 tester.setAge(age);  
  
 System.*out*.println("Write salary of Tester");  
 int salary = in.nextInt();  
 tester.setSalary(salary);  
  
 System.*out*.println("Write type of testing (QA | engineer)");  
 String typeTesting = in.next();  
 tester.setType(typeTesting);  
  
 String INSERT1 = "INSERT INTO tester(project\_id, salary, typetester, firstname, secondname) VALUES(?, ?, ?, ?, ?)";  
 PreparedStatement statement = connection.prepareStatement(INSERT1);  
 System.*out*.println("Select Project ID: ");  
 int id = in.nextInt();  
 statement.setInt(1, id);  
 statement.setInt(2, tester.getSalary());  
 statement.setString(3, tester.getType());  
 statement.setString(4, tester.getFirstName());  
 statement.setString(5, tester.getSecondName());  
  
 int excecuteUpdate = statement.executeUpdate();  
 }  
  
 } catch (SQLException ex) {  
 ex.printStackTrace();  
 }  
 }  
 else if(choice == 2)  
 {  
 try{  
 Connection connection = DriverManager.*getConnection*(URL, username, password);  
 Project project = new Project();  
  
 System.*out*.println("Write title of project: ");  
 String projectTitle = in.next();  
 project.setTitle(projectTitle);  
  
 System.*out*.println("Write direct and indirect costs of Project:");  
  
 System.*out*.println("Direct: ");  
 int directCost = in.nextInt();  
 project.setDirectCost(directCost);  
  
 System.*out*.println("Indirect: ");  
 int indirectCost = in.nextInt();  
 project.setIndirectCost(indirectCost);  
  
 String INSERT = "INSERT INTO project(title, directcost, indirectcost) VALUES(?, ?, ?)";  
  
 PreparedStatement statement = connection.prepareStatement(INSERT);  
 statement.setString(1, project.getTitle());  
 statement.setInt(2, (int)project.getDirectCost());  
 statement.setInt(3, (int)project.getIndirectCost());  
  
 int executeUpdate = statement.executeUpdate();  
 }  
 catch(SQLException exception)  
 {  
 exception.printStackTrace();  
 }  
 }  
 }  
  
 public static void select() {  
 try {  
 Scanner in = new Scanner(System.*in*);  
 String URL = "jdbc:postgresql://localhost:5432/postgres";  
 String username = "postgres";  
 String password = "1245emer";  
  
 Connection connection = DriverManager.*getConnection*(URL, username, password);  
 Statement statement = connection.createStatement();  
  
 String SELECT = "SELECT \* from project";  
 ResultSet resultProject = statement.executeQuery(SELECT);  
  
 System.*out*.println("Projects in DATABASE: ");  
 while(resultProject.next())  
 {  
 int projectId = resultProject.getInt(1);  
 String title = resultProject.getString(2);  
 int directCost = resultProject.getInt(3);  
 int indirectCost = resultProject.getInt(4);  
  
 System.*out*.println("ID: " + projectId);  
 System.*out*.println("Title: " + title);  
 System.*out*.println("Direct cost: " + directCost);  
 System.*out*.println("Indirect cost: " + indirectCost);  
 System.*out*.println();  
 }  
  
 System.*out*.println();  
  
 String SELECTCEO = "SELECT \* FROM ceo";  
 ResultSet resultCEO = statement.executeQuery(SELECTCEO);  
  
 while(resultCEO.next())  
 {  
 int ceoId = resultCEO.getInt(1);  
 int projectId = resultCEO.getInt(2);  
 String firstName = resultCEO.getString(3);  
 String secondName = resultCEO.getString(4);  
 int age = resultCEO.getInt(5);  
 int salary = resultCEO.getInt(6);  
  
 System.*out*.println("CEO ID: " + ceoId);  
 System.*out*.println("Project ID: " + projectId);  
 System.*out*.println("First Name: " + firstName);  
 System.*out*.println("Second Name: " + secondName);  
 System.*out*.println("Age: " + age);  
 System.*out*.println("Salary: " + salary);  
 System.*out*.println();  
 }  
  
 String SELECTDEVELOPER = "SELECT \* FROM developer";  
 ResultSet resultDeveloper = statement.executeQuery(SELECTDEVELOPER);  
  
 while(resultDeveloper.next())  
 {  
 int developerId = resultDeveloper.getInt(1);  
 int projectId = resultDeveloper.getInt(2);  
 int salary = resultDeveloper.getInt(3);  
 String category = resultDeveloper.getString(4);  
 String developerside = resultDeveloper.getString(5);  
 String firstName = resultDeveloper.getString(6);  
 String secondName = resultDeveloper.getString(7);  
  
 System.*out*.println("Developer ID: " + developerId);  
 System.*out*.println("Project ID: " + projectId);  
 System.*out*.println("First Name: " + firstName);  
 System.*out*.println("Second Name: " + secondName);  
 System.*out*.println("Salary: " + salary);  
 System.*out*.println("Category: " + category);  
 System.*out*.println("Developer side: " + developerside);  
 System.*out*.println();  
 }  
  
 String SELECTTESTER = "SELECT \* FROM tester";  
 ResultSet resultTester = statement.executeQuery(SELECTTESTER);  
  
 while(resultTester.next())  
 {  
 int testerId = resultTester.getInt(1);  
 int projectId = resultTester.getInt(2);  
 int salary = resultTester.getInt(3);  
 String typetester = resultTester.getString(4);  
 String firstName = resultTester.getString(5);  
 String secondName = resultTester.getString(6);  
  
 System.*out*.println("Tester ID: " + testerId);  
 System.*out*.println("Project ID: " + projectId);  
 System.*out*.println("First Name: " + firstName);  
 System.*out*.println("Second Name: " + secondName);  
 System.*out*.println("Salary: " + salary);  
 System.*out*.println("Tester type: " + typetester);  
 System.*out*.println();  
 }  
 }  
  
  
 catch(SQLException exception)  
 {  
 exception.printStackTrace();  
 }  
 }  
  
 public static void delete()  
 {  
 Scanner in = new Scanner(System.*in*);  
 String URL = "jdbc:postgresql://localhost:5432/postgres";  
 String username = "postgres";  
 String password = "1245emer";  
  
 try  
 {  
 Connection connection = DriverManager.*getConnection*(URL, username, password);  
 System.*out*.println("Table name? (ceo, developer, tester) ");  
 String tableName = in.nextLine();  
 System.*out*.println("Write an ID of employee: ");  
 int id = in.nextInt();  
 if(tableName.equals("ceo"))  
 {  
 String DELETE = "DELETE from ceo where ceo\_id = ?";  
 PreparedStatement statement = connection.prepareStatement(DELETE);  
 statement.setInt(1, id);  
 }  
 else if(tableName.equals("developer"))  
 {  
 String DELETE = "DELETE from developer where developer\_id = ?";  
 PreparedStatement statement = connection.prepareStatement(DELETE);  
 statement.setInt(1, id);  
 }  
 else if(tableName.equals("tester"))  
 {  
 String DELETE = "DELETE from tester where tester\_id = ?";  
 PreparedStatement statement = connection.prepareStatement(DELETE);  
 statement.setInt(1, id);  
  
 }  
 }  
 catch(SQLException exception)  
 {  
 exception.printStackTrace();  
 }  
 }  
  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 String URL = "jdbc:postgresql://localhost:5432/postgres";  
 String username = "postgres";  
 String password = "1245emer";  
  
 System.*out*.println("This program for IT Project. Our menu: ");  
 boolean key = false;  
 int decision = 0;  
 while (decision > 0 && decision < 5 || !key) {  
 try {  
 System.*out*.println("1. INSERT INTO");  
 System.*out*.println("2. SELECT");  
 System.*out*.println("3. DELETE");  
 System.*out*.println("4. QUIT");  
 System.*out*.println();  
 decision = in.nextInt();  
 switch (decision)  
 {  
 case 1:  
 *insert*();  
 break;  
 case 2:  
 *select*();  
 break;  
 case 3:  
 *delete*();  
 break;  
 case 4:  
 return;  
 default:  
 System.*out*.println("Incorrect number was inputted, write correct number (1-5)");  
 }  
 key = true;  
 } catch (InputMismatchException e) {  
 System.*out*.println("Please input an integer");  
 System.*out*.println("Required int, instead of: " + in.nextLine()); //for retarding wrong inputten data used in.nextLine()  
 }  
 }  
 }  
}