**import** jakarta.persistence.\*;  
**import** lombok.\*;  
**import** java.time.LocalDate;  
**import** java.time.LocalDateTime;  
  
@Entity  
@Table(name = "users")  
@Data  
@AllArgsConstructor  
@NoArgsConstructor  
**public class** User {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** String fullName;  
 **private** String email;  
 **private** String password;  
 **private** String address;  
 **private** String gender;  
 **private** LocalDate dateOfBirth;  
 **private** String nationalId;  
 **private** String contact;  
 **private double** basicSalary;  
 **private** LocalDateTime joinedDate;  
 **private boolean** isActive;  
 **private** LocalDateTime createdAt;  
 **private** LocalDateTime updatedAt;  
 **private** String profilePhoto;  
  
  
 @Enumerated(EnumType.*STRING*)  
 **private** Role role;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "companyId")  
 **private** Company company;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "departmentId")  
 **private** Department department;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "branchId")  
 **private** Branch branch;  
  
 @OneToMany(mappedBy = "user", fetch = FetchType.*LAZY*, cascade = CascadeType.*PERSIST*)  
 @JsonIgnore  
 **private** List<Attendance> attendances;  
  
}

**public enum** Role {  
 *ADMIN*,  
 *COMPANY*,  
 *MANAGER*,  
 *PAYROLLADMIN*,  
 *EMPLOYEE*}

**public class** Attendance {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** LocalDate date;  
 **private** LocalDateTime clockInTime;  
 **private** LocalDateTime clockOutTime;

**private int** year;

@Enumerated(EnumType.*STRING*)  
 **private** Month attendanceMonth;  
   
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "userId", nullable = **false**)  
 **private** User user;  
}

**public enum** Month {  
 *JANUARY*,  
 *FEBRUARY*,  
 *MARCH*,  
 *APRIL*,  
 *MAY*,  
 *JUNE*,  
 *JULY*,  
 *AUGUST*,  
 *SEPTEMBER*,  
 *OCTOBER*,  
 *NOVEMBER*,  
 *DECEMBER*}

**public class** Leave {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** LocalDate startDate;  
 **private** LocalDate endDate;  
 **private** LocalDateTime requestDate;  
 **private int** remainingLeave; // total 25 days: (reserve 10 + sick 15 days), remainingcalculation(total -(end-startdate))

**private int** year;

@Enumerated(EnumType.*STRING*)  
 **private** Month leaveMonth;

@Enumerated(EnumType.*STRING*)  
 **private** LeaveType leaveType;  
  
 @Enumerated(EnumType.*STRING*)  
 **private** RequestStatus requestStatus;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "userId", nullable = **false**)  
 **private** User user;  
}

**public enum** LeaveType {  
 *SICK\_PAID*,  
 *SICK\_UNPAID*,  
 *RESERVE\_UNPAID*}

**public enum** RequestStatus {  
 *PENDING*,  
 *APPROVED*,  
 *REJECTED*;  
}

**public class** AdvanceSalary {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private double** advanceSalary;  
 **private** String reason;  
 **private** LocalDateTime advanceDate;

**private int** year;

@Enumerated(EnumType.*STRING*)  
 **private** Month advanceMonth;

@ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "userId")  
 **private** User user;  
}

**public class** Bonus {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private double** bonusAmount;  
 **private** LocalDateTime bonusDate;

**private int** year;

@Enumerated(EnumType.*STRING*)  
 **private** Month bonusMonth;

@ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "userId")  
 **private** User user;  
  
  
}

**public class** Payroll {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** LocalDateTime paymentDate;  
 **private double** medicare;  
 **private double** providentFund; // baseSalary \* (2% = 0.02)  
 **private double** insurance;

**private double** transportAllowance;

**private double** telephoneSubsidy;

**private double** utilityAllowance;

**private double** domesticAllowance;

**private double** lunchAllowance;  
 **private boolean** overTime; //(If Yes then add 1days salary=(baseSalary/4 week /5 days)

**private double** netSalary;

**private int** year;

@Enumerated(EnumType.*STRING*)  
 **private** Month payrollMonth;

**private double** tax;

@ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "userId")  
 **private** User user;  
  
 @ManyToOne(fetch = FetchType.*LAZY,* cascade = CascadeType.PERSIST)  
 @JoinColumn(name = "advancesalaryId")  
 **private** AdvanceSalary advanceSalary;  
  
 @OneToMany (fetch = FetchType.*LAZY,* cascade = CascadeType.PERSIST)  
 @JoinColumn(name = "bonusId")  
 **private** List<Bonus> bonuses;  
  
 @OneToMany (fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "leaveId")  
 **private** List<Leave> leaves;  
   
}

**public class** Feedback {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** LocalDateTime date;  
 **private** String rating;  
 **private** String comment;

@Enumerated(EnumType.*STRING*)  
 **private** Month feedbackMonth;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "userId")  
 **private** User user;  
  
  
}

**public class** Company {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private** Long id;  
 **private** String name;  
 **private** String photo;  
   
 @OneToMany(mappedBy = "company", fetch = FetchType.*LAZY*, cascade = CascadeType.*PERSIST*)  
 @JsonIgnore  
 **private** List<Branch> branches;  
  
 @OneToMany(mappedBy = "company", fetch = FetchType.*LAZY*, cascade = CascadeType.*PERSIST*)  
 @JsonIgnore  
 **private** List<Department> departments;  
  
 @OneToMany(mappedBy = "company", fetch = FetchType.*LAZY*)  
 @JsonIgnore  
 **private** List<User> users;  
  
  
}

**public class** Branch {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** String name;  
 **private** String address;  
 **private** String city;  
 **private** String zipCode;  
 **private** String country;  
 **private** LocalDateTime createdAt;  
 **private** LocalDateTime updateAt;  
 **private** String photo;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "companyId")  
 **private** Company company;  
  
 @OneToMany(mappedBy = "branch", fetch = FetchType.*LAZY*)  
 @JsonIgnore  
 **private** List<User> users;  
  
}

**public class** Department {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 **private long** id;  
 **private** String name;  
 **private int** employeesNum;

**private** String photo;

@ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "branchId")  
 **private** Branch branch;  
  
 @ManyToOne(fetch = FetchType.*LAZY*)  
 @JoinColumn(name = "companyId")  
 **private** Company company;  
  
 @OneToMany(mappedBy = "department", fetch = FetchType.*LAZY*)  
 @JsonIgnore  
 **private** List<User> users;  
  
}