



# **LẬP TRÌNH JAVA**

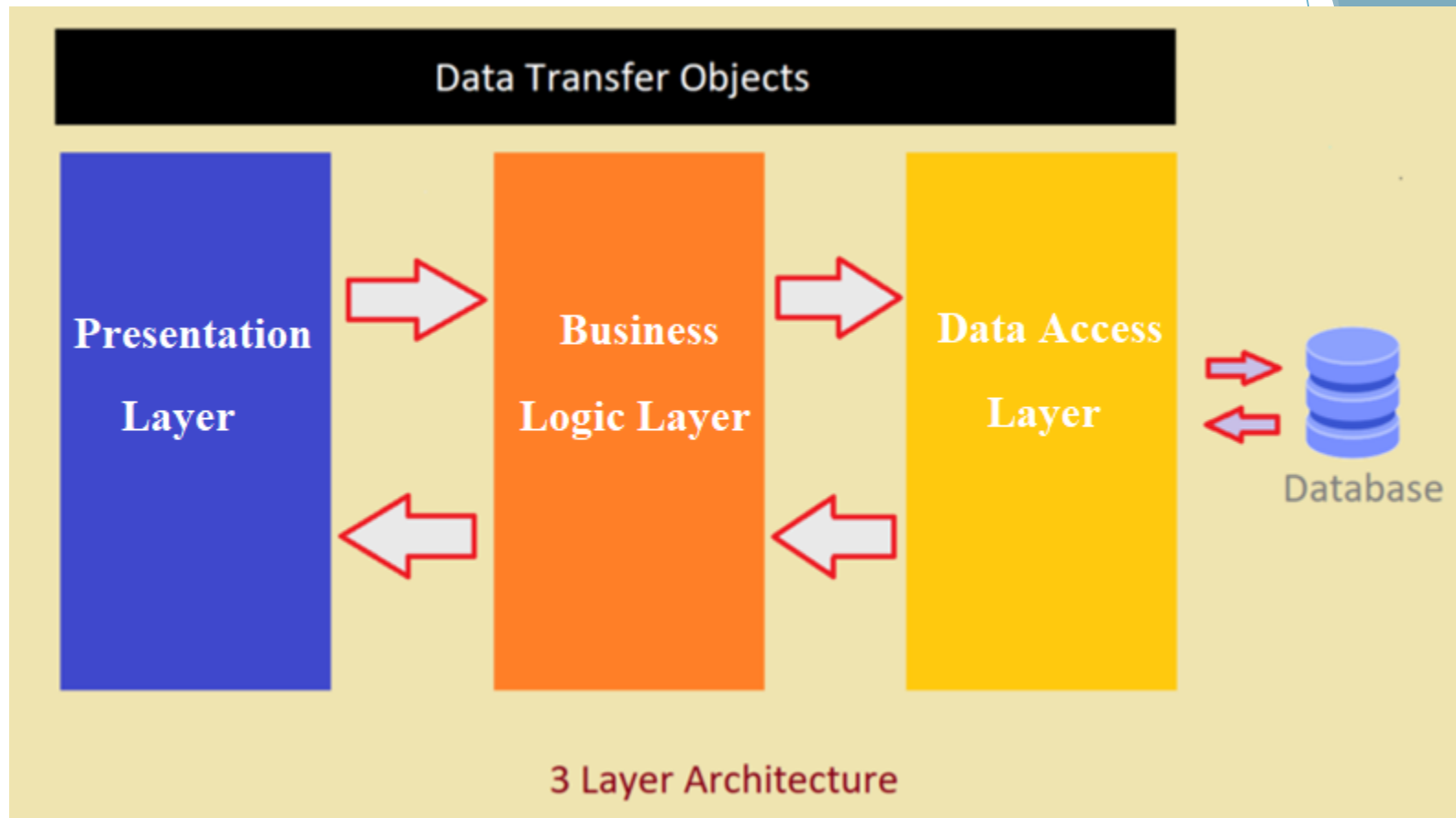
## **Mô hình 3 lớp**

**Nguyễn Thị Hồng Anh – Đỗ Ngọc Như Loan**

## Giới thiệu mô hình 3 lớp (3-layer)

- ▶ Trong phát triển ứng dụng, mô hình 3 lớp ra đời nhằm phân chia các thành phần trong hệ thống để dễ quản lý.
- ▶ Các thành phần cùng chức năng sẽ được nhóm lại với nhau và phân chia trách nhiệm cho từng nhóm để công việc không bị chồng chéo và ảnh hưởng lẫn nhau.

# Giới thiệu mô hình 3 lớp (3-layer)



# Presentation Layer (GUI)

- ▶ Lớp này có nhiệm vụ chính là giao tiếp với người dùng.
- ▶ Gồm các thành phần giao diện và thực hiện các công việc như nhập liệu, hiển thị dữ liệu, kiểm tra tính đúng đắn dữ liệu trước khi gọi lớp Business Logic Layer.

# Business Logic Layer (BLL/BUS)

- ▶ Xử lý dữ liệu từ Presentation Layer trước khi truyền xuống Data Access Layer và lưu xuống hệ quản trị CSDL.
- ▶ Kiểm tra các ràng buộc, tính toàn vẹn và hợp lệ dữ liệu, thực hiện tính toán và xử lý các yêu cầu nghiệp vụ, trước khi trả kết quả về Presentation Layer.

# Data Access Layer (DAL/DAO)

- ▶ Lớp này có chức năng giao tiếp với hệ quản trị CSDL
- ▶ Thực hiện các công việc liên quan đến lưu trữ và truy xuất dữ liệu (đọc, thêm, xóa, sửa,...)

# Data Transfer Object (DTO)

- ▶ Các layer trao đổi dữ liệu thông qua Data Transfer Object (DTO).
- ▶ DTO là đối tượng chứa dữ liệu, được tạo từ các DTO class đại diện cho các đối tượng được lưu trữ trong CSDL.

# Cách thức vận hành

- ▶ Đầu tiên người dùng giao tiếp với Presentation Layer (GUI) để gửi thông tin yêu cầu.
- ▶ Tại GUI, các thông tin sẽ được kiểm tra sơ bộ, nếu hợp lệ sẽ được chuyển xuống Business Logic Layer (BLL).
- ▶ BLL sẽ kiểm tra và tính toán các yêu cầu nghiệp vụ, nếu yêu cầu không hợp lệ hoặc không cần đến database thì BLL sẽ trả kết quả về cho GUI.



# Cách thức vận hành

- ▶ Ngược lại, BLL sẽ gửi yêu cầu (đã xử lý nghiệp vụ) xuống Data Access Layer (DAL).
- ▶ DAL sẽ thao tác với hệ quản trị CSDL và trả kết quả về cho BLL.
- ▶ BLL kiểm tra và gửi nó lên GUI.
- ▶ GUI sẽ hiển thị thông báo và kết quả yêu cầu cho người dùng.


# Ưu điểm của mô hình 3 lớp

- ▶ Giúp code được tường minh hơn, giảm sự kết dính nhờ vào việc chia ra từng lớp đảm nhận các chức năng khác nhau và riêng biệt như giao diện, xử lý, truy vấn.
- ▶ Dễ bảo trì: Khi 1 thành phần của hệ thống thay đổi, việc thay đổi này có thể được cô lập trong 1 lớp, hoặc ảnh hưởng đến lớp gần nhất mà không ảnh hưởng đến cả chương trình.

# Ưu điểm của mô hình 3 lớp

- ▶ Dễ phát triển: Khi muốn thêm chức năng nào đó thì việc lập trình theo mô hình sẽ dễ dàng hơn vì đã có chuẩn để tuân theo.
- ▶ Dễ bàn giao: Việc bàn giao, tương tác với nhau sẽ dễ dàng, nhanh chóng hơn vì mọi người đều theo một quy chuẩn định sẵn.
- ▶ Dễ phân phối khối lượng công việc: Mỗi nhóm sẽ nhận 1 nhiệm vụ được phân chia rõ ràng trong mô hình 3 lớp.

# Ví dụ

 Ví dụ mô hình 3 lớp

Mã NV	Họ tên	Năm sinh	Địa chỉ
1000	Mai	1990	quan 7
1001	Minh	1992	quan 1
1002	Lan	2000	quan 3

Mã NV

Họ tên


Năm sinh

Địa chỉ

Thêm

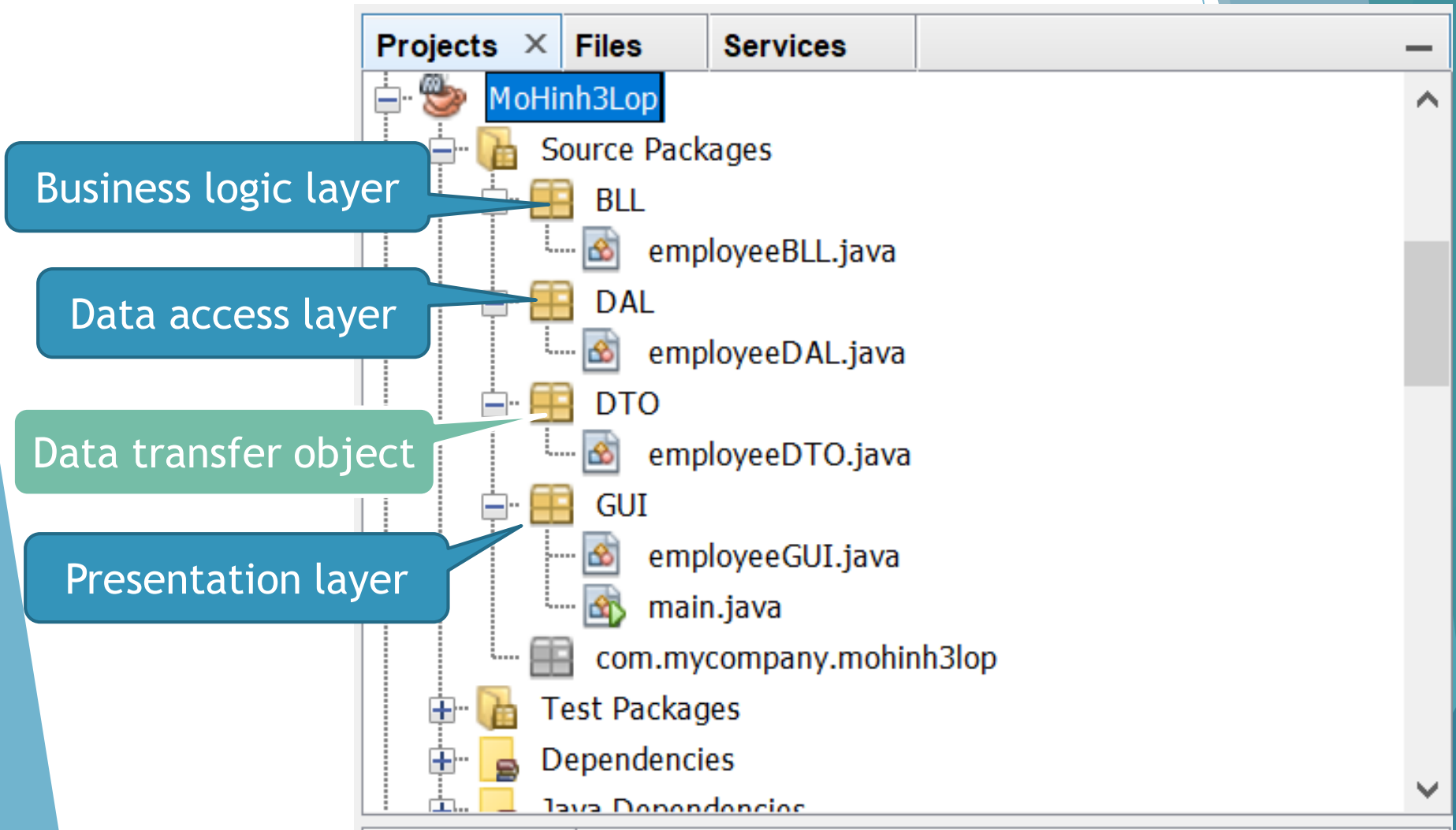
# Ví dụ - CSDL

CSDL QLNhanSu có bảng Employee với cấu trúc như sau:

	Column Name	Data Type
	employee_id	int
	employee_name	nvarchar(50)
	birthyear	int
	address	nvarchar(100)



# Ví dụ - Java project – mô hình 3 lớp



# Ví dụ - Data transfer object (employeeDTO.java) (1)



```
package DTO;

public class employeeDTO {
    private int employee_id;
    private String employee_name;
    private int birthyear;
    private String address;
    public int getEmployee_id() {
        return employee_id;    }
    public void setEmployee_id(int employee_id) {
        this.employee_id = employee_id;    }
    public String getEmployee_name() {
        return employee_name;    }
```



## Ví dụ - Data transfer object (employeeDTO.java) (2)

```
public void setEmployee_name(String employee_name) {  
    this.employee_name = employee_name; }  
public int getBirthYear() {  
    return birthyear; }  
public void setBirthYear(int birthyear) {  
    this.birthyear = birthyear; }  
public String getAddress() {  
    return address; }  
public void setAddress(String address) {  
    this.address = address; }  
}
```



# Ví dụ - Data access layer (employeeDAL.java) (1)



```
package DAL;

import DTO.employeeDTO;
import java.sql.*;
import java.util.Vector;

public class employeeDAL {
    private Connection con;

    public boolean openConnection() {
        try {
            Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
            String dbUrl = "jdbc:sqlserver://localhost:1433;
DatabaseName=QLNhanSu";
            String username = "sa"; String password= "sa";
            con = DriverManager.getConnection(dbUrl, username, password);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

## Ví dụ - Data access layer (employeeDAL.java) (2)

```
        return true;
    } catch (Exception ex) {
        System.out.println(ex);
        return false; }
}

public void closeConnection() {
    try {
        if (con!=null)
            con.close();
    } catch (SQLException ex) {
        System.out.println(ex); }
```

# Ví dụ - Data access layer (employeeDAL.java) (3)



```
public Vector<employeeDTO> getAllEmployees(){  
    Vector<employeeDTO> arr = new Vector<employeeDTO>();  
    if (openConnection()) {  
        try {  
            String sql = "Select * from Employee";  
            Statement stmt = con.createStatement();  
            ResultSet rs = stmt.executeQuery(sql);  
            while(rs.next()){  
                employeeDTO em = new employeeDTO();  
                em.setEmployee_id(rs.getInt("employee_id"));  
                em.setEmployee_name(rs.getString("employee_name"));  
                em.setBirthYear(rs.getInt("birthyear"));  
                em.setAddress(rs.getString("address"));  
            }  
        }  
    }  
}
```

# Ví dụ - Data access layer (employeeDAL.java) (4)



```
        arr.add(em);    }
    } catch (SQLException ex) {
        System.out.println(ex);
    } finally {
        closeConnection();
    } }
    return arr;
}

public boolean addEmployee(employeeDTO emp) {
    boolean result = false;
    if (openConnection()) {
        try {
            String sql = "Insert into Employee values(?,?,?,?)";
```

# Ví dụ - Data access layer (employeeDAL.java) (5)



```
PreparedStatement stmt = con.prepareStatement(sql);  
stmt.setInt(1, emp.getEmployee_id());  
stmt.setString(2, emp.getEmployee_name());  
stmt.setInt(3, emp.getBirthYear());  
stmt.setString(4, emp.getAddress());  
if (stmt.executeUpdate()>=1)  
    result = true;  
} catch (SQLException ex) {  
    System.out.println(ex);  
} finally{  
    closeConnection(); } }  
return result;
```

# Ví dụ - Data access layer (employeeDAL.java) (6)



```
public boolean hasEmployeeID(int id){  
    boolean result = false;  
    if (openConnection()) {  
        try {  
            String sql = "Select * from Employee where  
employee_id="+id;  
            Statement stmt = con.createStatement();  
            ResultSet rs = stmt.executeQuery(sql);  
            result = rs.next();  
        } catch (SQLException ex) {  
            System.out.println(ex);  
        } finally {    closeConnection(); }    }  
    return result;  
}
```



# Ví dụ - Business Logic Layer (employeeBLL.java) (1)

```
package BLL;  
  
import DAL.employeeDAL;  
import DTO.employeeDTO;  
import java.util.Vector;  
  
public class employeeBLL {  
    employeeDAL empDAL = new employeeDAL();  
    public Vector<employeeDTO> getAllEmployees(){  
        return empDAL.getAllEmployees();  
    }  
}
```



## Ví dụ - Business Logic Layer (employeeBLL.java) (2)

```
public String addEmployee(employeeDTO emp) {  
    if (empDAL.hasEmployeeID(emp.getEmployee_id()))  
        return "Mã NV đã tồn tại";  
    if (empDAL.addEmployee(emp))  
        return "Thêm thành công";  
    return "Thêm thất bại";  
}  
}
```



# Ví dụ - Presentation layer (employeeGUI.java) (1)



```
package GUI;

import BLL.employeeBLL;    import DTO.employeeDTO;
import java.awt.*;          import java.awt.event.*;
import javax.swing.*;       import java.util.Vector;
import javax.swing.table.DefaultTableModel;

public class employeeGUI extends JFrame {
    employeeBLL empBLL = new employeeBLL();
    JTable jTable1;
    ... // khai báo các thành phần giao diện khác
    public employeeGUI() {
        initComponents();
        loadEmployeeList();
    }
}
```



## Ví dụ - Presentation layer (employeeGUI.java) (2)

```
public void initComponents() {  
    ..... // khởi tạo các thành phần giao diện  
}  
  
public void loadEmployeeList(){  
    DefaultTableModel dtm = new DefaultTableModel();  
    dtm.addColumn("Mã NV");  
    dtm.addColumn("Ho tên");  
    dtm.addColumn("Năm sinh");  
    dtm.addColumn("Địa chỉ");  
    jTable1.setModel(dtm);  
    Vector<employeeDTO> arr = new  
    Vector<employeeDTO>();
```

## Ví dụ - Presentation layer (employeeGUI.java) (3)

```
arr = empBLL.getAllEmployees();  
for(int i = 0; i < arr.size(); i++){  
    employeeDTO em = arr.get(i);  
    int id = em.getEmployee_id();  
    String name = em.getEmployee_name();  
    int birthyear = em.getBirthYear();  
    String address = em.getAddress();  
    Object[] row = {id,name,birthyear,address};  
    dtm.addRow(row);  
}  
}
```



## Ví dụ - Presentation layer (employeeGUI.java) (4)

```
public void addEmployeeActionPerformed(ActionEvent e) {  
    try {  
        if (txtID.getText().trim().equals("") ||  
            txtName.getText().trim().equals("") ||  
            txtYear.getText().trim().equals("") ||  
            txtAddr.getText().trim().equals(""))  
            JOptionPane.showMessageDialog(this, "Vui lòng nhập  
đủ thông tin");  
        else {  
            employeeDTO em = new employeeDTO();  
            em.setEmployee_id(Integer.parseInt(txtID.getText()));  
            em.setEmployee_name(txtName.getText());
```

## Ví dụ - Presentation layer (employeeGUI.java) (5)

```
        em.setBirthYear(Integer.parseInt(txtYear.getText()));
        em.setAddress(txtAddr.getText());
        JOptionPane.showMessageDialog(this,
empBLL.addEmployee(em));
        loadEmployeeList();
    }
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(this, "Thông tin không hợp lệ");
}
}
```



## Ví dụ - Presentation layer (main.java)

```
package GUI;  
  
public class main {  
    public static void main(String[] args){  
        employeeGUI frm = new employeeGUI();  
        frm.setVisible(true);  
    }  
}
```

# Bài tập

- ▶ Sử dụng mô hình 3 lớp cho bài tập **Đăng nhập** và **Đăng ký** ở chương trước.
- ▶ Làm đồ án nhóm (xây dựng ứng dụng theo mô hình 3 lớp, giao diện đồ họa và kết nối cơ sở dữ liệu).