### **1.Yêu cầu và mục tiêu**

Mục đích:Kiến trúc hệ thống của phần mềm

-Thu thập yêu cầu từ các bên liên quan (người dùng cuối, khách hàng, nhóm phát triển, v.v.).  
-Xác định các yêu cầu chức năng (chức năng của phần mềm) và phi chức năng (hiệu suất, bảo mật, mở rộng, v.v.).  
-Đảm bảo yêu cầu phải rõ ràng, đo lường được và có thể kiểm tra được.

**2. Thành phần chính của hệ thống**  
-Phân tích các thành phần của phần mềm : giao diện người dùng, logic nghiệp vụ, truy cập dữ liệu, API, tích hợp hệ thống  
-Xác định các dịch vụ, module hoặc microservices sẽ được sử dụng trong hệ thống.  
-Xác định các giao thức giao tiếp giữa các thành phần của hệ thống

### **3. Lựa chọn kiểu kiến trúc phần mềm**

Mục đích: Chọn mô hình kiến trúc phù hợp với yêu cầu và mục tiêu của phần mềm.  
Sử dụng Kiến trúc phân lớp : Phân chia thành Presentation Layer, Business Logic Layer, và Data Access Layer. . .

#### ***Các lớp trong kiến trúc***

**Lớp Người dùng (Presentation Layer)**:

-Mục đích: Đây là lớp chịu trách nhiệm giao tiếp với người dùng (UI - User Interface). Nó hiển thị dữ liệu và nhận các yêu cầu từ người dùng.

-Chức năng: Hiển thị thông tin, nhận input, chuyển đổi các yêu cầu từ người dùng thành các lời gọi tới các lớp dưới.

**-Cấu tạo**:**Giao diện người dùng (UI)**: Các thành phần giao diện người dùng như form, bảng, biểu đồ, và các nút điều khiển (button).

**Xử lý giao diện người dùng (UI Logic)**: Quản lý việc hiển thị các dữ liệu, xử lý sự kiện như nhấp chuột, kéo thả, nhập liệu.

**Chuyển tiếp yêu cầu**: Sau khi nhận dữ liệu từ người dùng, lớp này chuyển các yêu cầu này đến lớp ứng dụng hoặc lớp nghiệp vụ để xử lý.

**Công nghệ**: HTML, CSS, JavaScript, React, Angular, Vue.js (với ứng dụng web); JavaFX, Swing (với ứng dụng desktop)

**Lớp Ứng dụng (Application Layer)**:

Mục đích: Lớp này chịu trách nhiệm xử lý các yêu cầu nghiệp vụ, điều phối các lớp khác để thực hiện các công việc cụ thể.

Chức năng: Thực hiện các tác vụ nghiệp vụ, điều phối và gọi các phương thức từ lớp nghiệp vụ hoặc lớp dữ liệu.

**Cấu tạo**:

**Quản lý yêu cầu**: Lớp này nhận các yêu cầu từ lớp người dùng và chuyển tiếp tới lớp nghiệp vụ (Business Logic Layer) hoặc lớp dữ liệu (Data Layer).

**Dịch vụ ứng dụng**: Các dịch vụ xử lý logic đơn giản hoặc điều phối các yêu cầu từ lớp giao diện người dùng.

**Xử lý lỗi**: Lớp này cũng có thể xử lý các lỗi hoặc vấn đề phát sinh trong quá trình giao tiếp giữa các lớp khác.

**Chuyển tiếp thông tin**: Sau khi xử lý, lớp này chuyển kết quả trả về cho lớp người dùng để hiển thị cho người sử dụng.

**Công nghệ**: Spring Boot (Java), Express.js (Node.js), Django (Python), .NET.

**Lớp Nghiệp vụ (Business Logic Layer)**:

Mục đích: Lớp này chứa các quy tắc và logic nghiệp vụ của ứng dụng. Đây là nơi xử lý các yêu cầu nghiệp vụ phức tạp.

Chức năng: Xử lý các tính toán, điều kiện nghiệp vụ, và các tác vụ cần thiết cho hệ thống.

Cấu tạo:

Xử lý nghiệp vụ: Các phương thức và hàm thực hiện các tính toán, kiểm tra điều kiện, và thực thi các quy tắc nghiệp vụ.

Giao tiếp với lớp ứng dụng: Khi nhận yêu cầu từ lớp ứng dụng, lớp này thực hiện các tác vụ nghiệp vụ, có thể gọi đến lớp dữ liệu để lấy thông tin cần thiết.

Quản lý trạng thái nghiệp vụ: Lớp này quản lý trạng thái của các đối tượng nghiệp vụ trong suốt quá trình xử lý.

Công nghệ: Java (Spring), Python, Ruby on Rails, Node.js (với Express.js).

**Lớp Dữ liệu (Data Layer)**:

Mục đích: Lớp này chịu trách nhiệm lưu trữ và truy xuất dữ liệu từ các nguồn dữ liệu (database, API, file, v.v.).

Chức năng: Tương tác với cơ sở dữ liệu, thực hiện các thao tác CRUD (Create, Read, Update, Delete).

Cấu tạo:

Cơ sở dữ liệu (Database): Đây là nơi lưu trữ các dữ liệu ứng dụng như thông tin người dùng, sản phẩm, đơn hàng, v.v.

Đối tượng truy cập dữ liệu (DAO - Data Access Object): Là lớp trung gian giữa ứng dụng và cơ sở dữ liệu, thực hiện các thao tác CRUD (Create, Read, Update, Delete) đối với cơ sở dữ liệu.

ORM (Object-Relational Mapping): Công nghệ giúp ánh xạ đối tượng trong mã nguồn với các bảng trong cơ sở dữ liệu.

Tầng bảo mật dữ liệu: Quản lý việc bảo mật dữ liệu, kiểm tra quyền truy cập, mã hóa dữ liệu.

Công nghệ: MySQL, PostgreSQL, MongoDB, Hibernate (Java), SQLAlchemy (Python), Sequelize (Node.js).

**Lớp Hệ thống (System Layer - Optional)**:

Mục đích: Lớp này chịu trách nhiệm các công việc liên quan đến hạ tầng hệ thống như bảo mật, log hệ thống, hoặc xử lý bất đồng bộ.

Chức năng: Quản lý các tài nguyên hệ thống, thực hiện bảo mật, cấu hình hệ thống.

Cấu tạo:

Quản lý bảo mật: Quản lý xác thực và phân quyền người dùng, mã hóa thông tin nhạy cảm.

Log hệ thống: Lưu trữ các log về hoạt động của ứng dụng.

Cấu hình hệ thống: Cấu hình các tham số hệ thống như thời gian, phân vùng, hoặc môi trường chạy.

Công nghệ: Spring Security, Apache Kafka, RabbitMQ, JWT (JSON Web Tokens).

### **Mô hình hoạt động của Kiến trúc phân lớp**

Lớp người dùng (Presentation Layer) nhận các yêu cầu từ người dùng.

Yêu cầu sau đó được chuyển đến Lớp ứng dụng (Application Layer), nơi mà yêu cầu được xử lý và chuyển đến Lớp nghiệp vụ (Business Logic Layer).

Lớp nghiệp vụ (Business Logic Layer) thực hiện các công việc liên quan đến nghiệp vụ, sau đó yêu cầu dữ liệu từ Lớp dữ liệu (Data Layer).

Lớp dữ liệu (Data Layer) truy vấn cơ sở dữ liệu và trả kết quả lại cho Lớp nghiệp vụ (Business Logic Layer).

Kết quả cuối cùng được chuyển trở lại Lớp ứng dụng (Application Layer) và cuối cùng trả về Lớp người dùng (Presentation Layer) để hiển thị kết quả cho người dùng.

