- Java Core:

• các tính chất của OOP: Các tính chất của OOP trong Java bao gồm: đóng gói (encapsulation), kế thừa (inheritance), đa hình (polymorphism), và tính trừu tượng (abstraction).

• Hiểu về các access modifier: C ó 4 access modifier trong Java: public, private, protected, và default. Public cho phép truy cập từ bất kỳ đâu. Private cho phép truy cập trong cùng một lớp. Protected cho phép truy cập từ cùng một lớp hoặc lớp con. Default cho phép truy cập từ cùng một package.

• phân biệt Overload và Override: Overloading trong Java được xác định bởi sự hiện diện của nhiều phương thức có cùng tên nhưng các đối số khác nhau trong cùng một lớp. Override là việc định nghĩa lại một phương thức đã được kế thừa từ lớp cha.

• Abstract class và Interface :  một lớp Abstract là một lớp mà không có thể được khởi tạo và chứa các phương thức và biến mà không có thể được khởi tạo. Interface là một tập hợp các phương thức được định nghĩa dưới dạng một tập hợp các phương thức rỗng. Giao diện được sử dụng để tạo ra một bộ khung tương tác giữa các lớp. Giao diện không thể được khởi tạo và có chứa các hằng số.

• mô tả về == và equals: == dùng để so sánh giá trị của hai biến trong Java. Nếu giá trị của hai biến đều bằng nhau thì == sẽ trả về true. Trong khi đó, equals() là một phương thức của lớp Object được sử dụng để so sánh giá trị của hai đối tượng trong Java. Nó sẽ so sánh xem hai đối tượng có phải là cùng một đối tượng hay không.

• static và final khi sử dụng với thuộc tính và phương thức: Static: Khi khai báo một thuộc tính hoặc phương thức là static, nó có nghĩa là thuộc tính hoặc phương thức đó thuộc về lớp, không thuộc về một đối tượng cụ thể. Do đó, bạn có thể truy cập nó bằng tên lớp, không cần tạo ra một đối tượng cụ thể của lớp đó.

Final: Một thuộc tính hoặc phương thức được đánh dấu là final đã được khóa, nghĩa là không thể thay đổi giá trị của nó. Nó cũng có nghĩa là bạn không thể override phương thức final trong java.

• Biết cách xử lý về các Exception và một vài chú ý khi xử lý nhiều Exception cùng lúc:

Khi x ử lý nhiều exception cùng lúc, áp dụng các biện pháp sau đây:

* Tạo một cấu trúc điều khiển try-catch để phân loại các exception riêng biệt.
* Tạo một ngoại lệ gốc để bao hàm các ngoại lệ khác.
* Nên sử dụng tên cuối cùng của exception để xác định loại ngoại lệ.
* Sử dụng các phương thức để xử lý các exception trong cùng một thời điểm.
* Tạo một đối tượng để quản lý các exception.
* Ghi lại các exception bằng cách sử dụng các lớp cổ điển.
* Cẩn thận khi sử dụng finally để xử lý các exception. Đây là một phần quan trọng của quá trình xử lý exception.

- Spring:

• định nghĩa của Bean và IoC trong java? Bean và IOC trong java: Bean là một đối tượng được tạo bởi một container để quản lý các đối tượng của chương trình. IOC là một phương pháp để quản lý đối tượng bằng cách sử dụng container để quản lý sự tạo ra, quản lý hoạt động và phân bổ các đối tượng.

• cách hoạt động của DI : DI trong Java: DI (Dependency Injection) là một phương pháp để quản lý phụ thuộc giữa các thành phần. Nó theo chủ đề “Kiểm soát kéo”, trong đó các thành phần có thể được giao bởi một container hoặc bởi một thành phần khác.

• ORM : ORM trong Java: ORM (Object Relational Mapping) là một kỹ thuật cho phép bạn xây dựng các đối tượng tương ứng với các mô hình dữ liệu của bạn. Nó cũng được gọi là một cách để chuyển đổi dữ liệu từ một dạng này sang dạng khác.

• Liệt kê các Http Method : Liệt kê các HTTP Method trong Java: Các phương thức HTTP trong Java bao gồm GET, POST, PUT, DELETE, HEAD, OPTIONS, CONNECT, TRACE và PATCH.

• lấy dữ liệu từ request thông qua RequestParam, PathVariable và RequestBody : Lấy dữ liệu từ Request thông qua RequestParam, PathVariable và RequestBody trong Java: RequestParam là một annotation được sử dụng để lấy dữ liệu từ tham số truyền lên bằng định dạng URL. PathVariable là một annotation được sử dụng để lấy dữ liệu từ tham số truyền lên bằng định dạng URL. RequestBody là một annotation được sử dụng để lấy dữ liệu từ tham số truyền lên bằng định dạng dữ liệu json.

• JWT : JWT (JSON Web Token) là gì trong java? JWT là một trong những thực thể cung cấp cho các ứng dụng web và di động một cơ chế bảo mật phức tạp được sử dụng để xác thực người dùng và chia sẻ thông tin với mục đích phân quyền.

• thao tác với cơ sở dữ liệu : Các thao tác cơ bản với cơ sở dữ liệu trong My SQL bao gồm tạo cơ sở dữ liệu, tạo bảng, chèn dữ liệu, cập nhật dữ liệu, xóa dữ liệu, sử dụng cú pháp SQL và thực thi câu lệnh.

- SQL:

1. • group by , mô tả về having: Group By: Group by là một cú pháp SQL được sử dụng để tạo ra các nhóm với các bản ghi trong một kết quả truy vấn cụ thể. Nó sẽ sắp xếp dữ liệu theo cột nào đó và gom nhóm các bản ghi thỏa mãn điều kiện nhất định của nó… Mô tả về Having trong My SQL: Having là một cấu trúc cú pháp SQL được sử dụng trong truy vấn của chúng ta để lọc các dòng kết quả dựa trên điều kiện cụ thể. Nó tương tự như một từ khóa WHERE, nhưng có thể được sử dụng với các hàm tính toán như sum, avg, count vv.
2. • cách hoạt động của join : Cách hoạt động của Join trong My SQL: Join là một cú pháp SQL được sử dụng để liên kết các bảng cơ sở dữ liệu trong một câu lệnh truy vấn. Điều này cho phép bạn lấy dữ liệu từ các bảng có liên quan nhau trong cùng một lệnh SQL.

• cú pháp insert/update : Cú pháp Insert/Update trong MySQL: INSERT INTO table\_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

UPDATE table\_name SET column1 = value1, column2 = value2, ... WHERE condition;

Sử dụng toán tử View, Store Procedure trong MySQL: View: CREATE VIEW view\_name AS SELECT column1, column2, ... FROM table\_name WHERE condition;

Store Procedure: CREATE PROCEDURE sp\_name [parameter\_list] BEGIN statements; END;

• sử dụng toán tử like : Sử dụng toán tử Like trong MySQL: SELECT column1, column2, ... FROM table\_name WHERE columnN LIKE pattern;

• KDL cơ bản trong SQL : KDL cơ bản trong SQL trong MySQL:

* SELECT/UPDATE/DELETE/INSERT
* JOINs
* Subqueries
* Views
* Stored Procedures
* Triggers
* Indexes

- Angular:

* • các cách truyền dữ liệu giữa các component
* @Input: truyền dữ liệu từ component cha đến con
* @Output: truyền dữ liệu từ component con đến cha
* Services: truyền dữ liệu giữa các component không liên quan
* Router: truyền dữ liệu giữa các component khi chuyển hướng

• các thành phần của component

Các thành phần của Component trong Angular:

* Template: định nghĩa giao diện của component
* Class: định nghĩa các thuộc tính và hành vi bên trong component
* Metadata: định nghĩa thông tin bên ngoài của component

• Biết cách sử dụng các component từ module khác

Các cách truyền dữ liệu giữa các component trong Angular:

Biết cách sử dụng các component từ module khác trong Angular: Sử dụng từ khóa import để sử dụng các component từ module khác. Ví dụ: import { ComponentName } from './moduleName';

• Interceptor

Interceptor trong Angular: Interceptor là một công nghệ cho phép chúng ta có thể thêm một số logic trước khi một request HTTP được gửi đi hoặc sau khi response được trả về. Interceptor cũng được sử dụng để có thể handling error của HTTP request.

- Giới thiệu kỹ hơn, tập trung vào một dự án lớn nhất, kỹ năng chuyên môn nắm chắc nhất.

- Hỏi về kiến thức cơ bản nhanh:

+ java cơ bản: kế thừa, access modifier, security, lamda, pojo

+ java nâng cao: hibernate: flush, transaction, các kiểu mapping, orm là gì. microservice(nếu nói biết), design pattern, web server và deploy

+ sql: join, group by, having, connect by (đệ quy), union, toán tử so sánh( nâng cao thì hỏi thêm có null). PL/SQL.

+ Với Lập trình web: - javascript (framework, ajax, post form..), - css: css3, bootstrap, responsive, html (một số thẻ cơ bản),

- công cụ hỗ trợ khi làm việc(đại khái khi cần cắt file photoshop sang html em cần công cụ gì. debug trên client bằng gì). - Hiểu về các loại request (post, get, put, options..) - Hiểu biết về attt: các lỗi attt, cors... - linux: ls, top, netstat, tail, vi, du ... (các lệnh dev hay phải dùng)