脳みそから煙が出そう。。

・どのようなプログラミング言語、フレームワーク、ツールなどをご経験されていますか？

・担当していたサービスの規模はどのくらいですか？

・以前のプロジェクトで担当していたポジション、業務は何ですか？

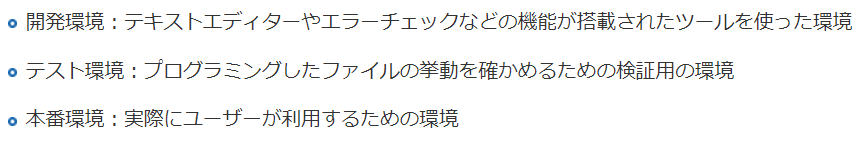
・どのような開発の進め方を経験してこられましたか？

オブジェクト指向プログラミング言語

リレーショナルデータベース（関係データベース、**RDB**）例：Oracleの「Oracle Database」、IBMの「IBM DB2」、マイクロソフトの「Microsoft SQL Server」、オープンソースベースの「MySQL」や「PostgreSQL」などがある。

Extensible Markup Language (**XML**) は、任意のデータを定義するルールを提供する**マークアップ言語**です。 他のプログラミング言語とは異なり、XML は単独では計算オペレーションを実行できません。 代わりに、構造化データ管理のために、任意のプログラミング言語またはソフトウェアを実装できます。

デプロイ　部署「deploy」「配置する」ITの分野では、システムにおいて「実行ファイルを実際のWebサーバー上に配置して、利用できる状態にすること」を指します。



プロバイダー

ビルド プログラミングファイルを実行ファイルへ置き換える作業

レンダリング 渲染

コードカバレッジ

インテグレーションテスト　結合テスト　Integration Test（IT）

リポジトリ（repository）

圧縮　あっしゅく

グローバル変数

プロパティ(property)

エレメントelement 要素

**ディレクティブ(vue)**

ディレクトリ: WindowsやMacなどの、いわゆるGUIでは「フォルダ」という呼び名を採用しています。対して、LinuxやUnixなどのCUIでは「ディレクトリ」と呼ばれます。

なぜ呼び方が違うのかというと、CUIにおいてはフォルダという概念が存在しないからです。

**疎結合(そけつごう)・密結合なコード** 疎結合とは個々のモジュールやコンポーネント同士が直接的に依存していない状態を指します。別のモジュールを呼び出す際に直接呼び出すのではなく、つなぎ目となるコードを経由して呼び出すことで疎結合な状態が実現されると言われます

OS operating system オペレーティングシステム 开发环境？

システムのTCO(Total Cost of Ownership)の低減

フェーズ 【phase】

COBOL

Solaris

Kotlin (Android)

Swift (iPhone)

Groovy

Subversion オープンソースのバージョン管理システム（VCS）の一つ

StarTeamはバージョン管理とソフトウェア構成管理の機能を持つソフトウェアである

Ruby on Rails

Terraform インフラストラクチャ ソフトウェア ツールです

VSS Microsoft Visual SourceSafe是美国微软公司出品的版本控制系统

**Struts**（web框架) オープン・ソース・ソフトウェアのフレームワークの 1 つで、 Web アプリケーションの作成に役立ちます。このフレームワークは、 Java™ Bean、Java サーブレット、JavaServer Pages (JSP)、XML などの標準テクノロジーを利用します。

**Azure**（アジュール）とは、マイクロソフト社が提供するパブリッククラウドのプラットフォーム

**ソフトウェアのクラウド化（SaaS）開発環境のクラウド化（PaaS）社内インフラのクラウド化（IaaS）**

サーブレット（Java **Servlet**）とは、Webサーバ上（バックエンド）で動くプログラムのことで、プログラミング言語のJavaを使って作成されています。 動的なWebページの配信システムを実現するためのJavaプログラムです。

**Tomcat** ソフトだよ// Java Servletを動かすときに使うよ // （簡易的な）Webサーバの機能も付いている

サーブレット を実行するためのWebコンテナである。

ストリングのメソッド string方法

Equals Length Trim ToUpper/LowerCase IndexOf

Spring Bootでよく使うアノテーション

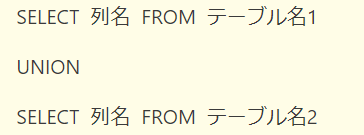
**SQL**

表连接

JOINとは、テーブル同士を結合することです

INNER JOIN(内部結合) 　テーブル同士を内部結合した場合、指定した条件に合致したレコードのみを取り出します。

LEFT JOIN(左外部結合)　　左テーブルのレコードすべてと、結合条件にマッチする右テーブルのレコードを返します。



「UNION」を使うことで、複数の検索結果を統合し1つの結果として取得することができます。

UNION------------**重複したレコードは1つのレコードとして統合**され

UNION ALL-------**重複したレコードも取得**

**Git常用命令**

**git branch issue1**

issue1という名前でブランチを作成

**git checkout (branchname)**

pullを実行すると、リモートリポジトリの内容のマージが自動的に行われてしまいます。しかし、単にリモートリポジトリの内容を確認したいだけの時はマージをしたくない場合もあります。そのような時はfetchを使用します。

fetchを実行すると、リモートリポジトリの最新の履歴の取得だけを行うことができます。

Pullリモートリポジトリからプル

pushリモートリポジトリにプッシュ

**リモートリポジトリをクローン**

（解决冲突）リモートリポジトリとローカルリポジトリでファイル内の同じ箇所を変更していた場合, どちらの変更を取り込むか自動では判断できない、すべての競合箇所を手動で修正してから、コミット

**Linux常用命令**

su ...今の環境を引き継いだまま、ユーザーを切り替え(ユーザー名を指定しないと、スーパーユーザーへの切り替えとなります)。

mkdir file

→ディレクトリfileを作成します。

AWS（Amazon Web Services）は、クラウドコンピューティングプロバイダーであり、さまざまなサービスを提供しています。以下にいくつかの代表的なAWSサービスを簡単に説明します：

EC2（Elastic Compute Cloud）：仮想サーバーを提供するサービスです。EC2を使用すると、インスタンス（仮想マシン）を作成し、アプリケーションの実行やデータの処理を行うことができます。

Amazon S3（Simple Storage Service）：オブジェクトストレージサービスで、データの保存やバックアップ、静的なウェブサイトのホスティングなどに使用されます。大容量のデータを安全に保存することができます。

Amazon RDS（Relational Database Service）：リレーショナルデータベースを提供するマネージドサービスです。MySQL、PostgreSQL、Oracleなどのデータベースエンジンを使用して、スケーラブルなデータベース環境を簡単にセットアップできます。

Lambda(ラムダ)：サーバーレスコンピューティングサービスで、コードを実行するための環境を提供します。Lambdaを使用すると、インフラ ストラクチャの管理をせずに、アプリケーションのコードを実行できます。

DynamoDB(ダイナモ ディービー)：高速でフルマネージドなNoSQLデータベースサービスです。柔軟なスケーラビリティ、低レイテンシ、高可用性を提供し、大規模なアプリケーションやリアルタイムデータの処理に適しています。

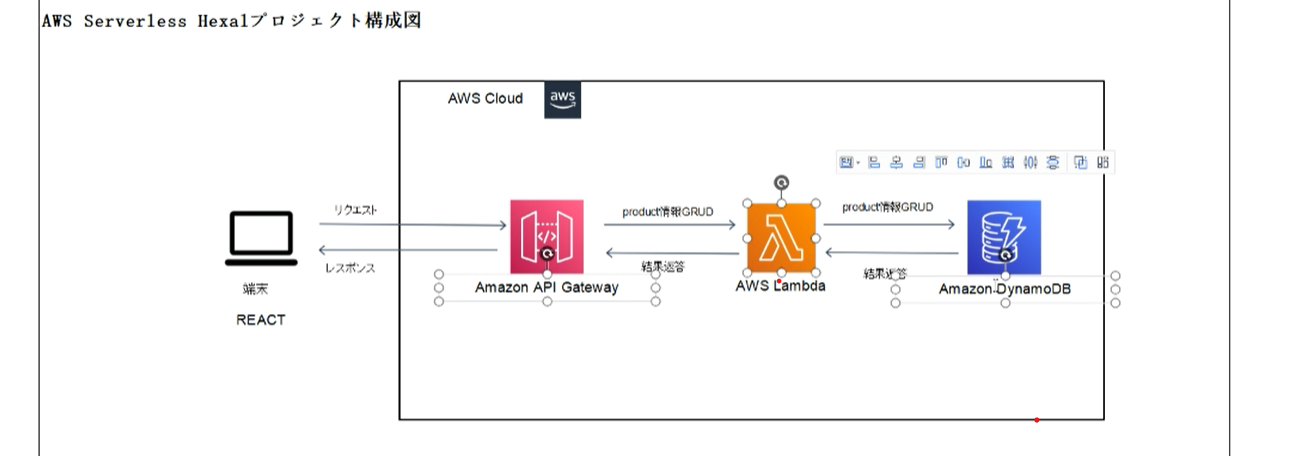
Amazon SQS（Simple Queue Service）：キューイングおよびメッセージングサービスで、分散アプリケーション間の非同期通信をサポートします。複数のコンポーネント間でメッセージを送受信することができます。

これらはAWSの一部ですが、実際にはさまざまなサービスがあります。AWSは、クラウドベースのインフラストラクチャとサービスを提供することで、スケーラビリティ、信頼性、セキュリティを向上させ、さまざまなビジネスニーズに対応します。

API gateway APIの作成および諸々の管理を行えるサービス。イメージとしてはリクエストの受口、そこからLambda関数を呼び出してバックエンドの処理を行う

API Gatewayでは、HTTP API または REST APIを使用して、[RESTful](https://aws.amazon.com/jp/what-is/restful-api/)なAPIを作成することができます。また[WebSocket API](https://developer.mozilla.org/ja/docs/Web/API/WebSockets_API)の作成も可能です。

无服务器架构



[axios](https://github.com/mzabriskie/axios)は、HTTP通信（データの更新・取得）を簡単に行うことができるJavascriptライブラリです。

**axiosの特徴**

* XML HttpReqestを簡単に生成できる
* Promiseベースである
* カスタムヘッダーやBasic認証など、いろいろなオプションが手軽にできる
* $ npm install axios　　　インストールは簡単で、npm経由で導入できます。

Vuex

Pinia

router

**ディレクティブ**とは**接頭辞 v- が付いたVue.jsの特別な属性のこと**を指します。

**---v-on**イベントリスナーを加えることで、Vue インスタンスのメソッドを呼び出すことができます。

**---v-bind 要素の属性を束縛(バインディング)**

---**v-model 双方向データバインディング**

**---v-for 配列に基づいた要素のリストレンダリング**

**---v-if 条件分岐によって、簡単に要素の有無の切り替え**

**v-ifとv-showの違い** v-ifとの**違い**は、**（v-showは）要素が常にDOMに保持される**ということです。つまり、v-showは要素の表示と非表示の切り替えをCSSの書き換えによって実現します。**使い分け**：v-ifは初期表示のコストが低い代わりに、SPA上で表示切り替えを行うためのコストが高いです。そして、v-showは初期表示のコストが高い代わりに、表示切り替えをCSSで行うため、表示の切り替えに要するコストが低いと言えます。つまり、  
**切り替えの頻度が低ければv-if**  
**切り替えの頻度が高ければv-show**

**CSS面试题**  
1.盒模型  
2.如何让一个盒子水平垂直居中？  
3.css 优先级确定  
4.解释下浮动和它的工作原理，清除浮动的方法？  
5.CSS隐藏元素的几种方法  
6.如何实现浏览器内多个标签页之间的通信?  
7.简要说一下CSS的元素分类  
8.link @import 导入 css  
9.画三角形  
10.BFC（Block Formatting Context） 是什么？应用？  
JavaScript基础面试题  
1.栈和堆的区别？  
2.Javascript实现继承的几种方式？  
3.Javascript创建对象的几种方式？  
4.Javascript作用链域  
5.什么是闭包（closure），为什么要用它？  
6. javascript 代码中的"use strict";是什么意思 ? 使用它区别是什么？  
7.深拷贝和浅拷贝  
8.JS延迟加载的方式有哪些？  
9.什么是跨域问题 ，如何解决跨域问题?  
10.模块化开发怎么做？  
11.DOM操作  
12.什么是Cookie 隔离？  
13.响应事件  
14.flash和js通过什么类如何交互?  
15.JS垃圾回收机制？  
Vue框架面试题  
1.对于MVVM的理解？  
2.Vue的生命周期  
3.Vue组件间的参数传递  
4.Vue实现数据双向绑定的原理  
5.Vue的路由实现：hash模式 和 history模式  
6.vue路由的钩子函数  
7.请问 v-if 和 v-show 有什么区别？  
8.对于Vue是一套渐进式框架的理解  
9.Vue 组件间通信有哪几种方式？  
10.vue中子组件调用父组件的方法  
11.怎么定义vue-router的动态路由？怎么获取传过来的值？  
12.请介绍一下你对vue-router的理解？  
13.vue-router有哪几种路由守卫?  
14.vuex有哪几种属性？  
15.vuex的State特性是？  
16.vue-cli如何新增自定义指令？  
17.vue等单页面应用及其优缺点  
浏览器面试题  
1.跨标签页通讯  
2.浏览器架构  
3.浏览器下事件循环(Event Loop)  
4.从输入 url 到展示的过程  
5.重绘与回流  
6.存储  
7.Web Worker  
8.V8 垃圾回收机制  
9.内存泄露  
服务端与网络  
1.http/https 协议  
2.常见状态码  
3.get/ post  
4.Websocket  
5.TCP 三次握手  
6.TCP 四次挥手  
7.Node的Event Loop: 6 个阶段  
8.跨域  
9.安全

Webpack 相关  
1.原理简述  
2.Loader  
3.Plugin  
4.编译优化  
算法  
1.五大算法  
2.基础排序算法  
3.高级排序算法  
4.递归运用(斐波那契数列)： 爬楼梯问题  
5.数据树  
6.天平找次品  
  
React框架  
1.Fiber  
2.生命周期  
3.setState  
4.HOC(高阶组件)  
5.Redux  
6.React Hooks  
7.SSR 服务端渲染  
8.函数式编程

Node.js 相关的可能包括：

1.为什么要用 Node.js（而不是 PHP/JAVA/GO/C++ 等）

2.Node.js 有哪些特点，单线程的优势和缺点是什么

3.Node.js 有哪些定时功能

4.Process.nextTick 和 setImmediate 的区别

5.Node.js 中的异步和同步怎么理解，异步流程如何控制

6.简单介绍一下 Node.js 中的核心内置类库(事件，流，文件，网络等)

7.Express 是如何从一个中间件执行到下一个中间件的