前端面试高频问题FAQ

✅ 核心原则：真诚不敷衍、正向不抱怨、贴合岗位、逻辑清晰，技术问题紧扣前端八股+项目实操

1. 面试候机法则（深入掌握）

- 提前30分钟到面试地点，不早到不迟到；
- 到现场后整理仪容、手机调静音，提前过一遍自我介绍/项目核心亮点；
- 主动核对面试信息，礼貌对接前台/HR，不随意走动、不闲聊；
- 候机时快速梳理岗位JD核心要求，预判面试官会问的技术/场景问题；
- 心态放平，不焦虑不慌神，保持自然松弛的状态。

2. 优秀自我介绍（前端岗模版，可直接套用）

通用版（1.5分钟，适配社招）

面试官好，我是XXX，有X年前端开发经验，主攻**Vue/React+TS**技术栈，熟练掌握小程序、H5、PC端项目开发，精通工程化、性能优化、跨端适配核心技能。
过往主导/参与过【XX电商小程序】【XX后台管理系统】【XX大屏可视化项目】，核心负责前端架构搭建、核心业务模块开发、性能调优，曾解决过【首屏加载慢/跨域/兼容性】等关键问题，落地过【按需加载/组件封装/接口统一管理】等优化方案，提升了项目效率和用户体验。
非常认可贵司的业务方向，也看好前端岗位的发展空间，希望能加入团队，用技术为业务赋能，谢谢面试官。

精简版（1分钟，适配初面/快速沟通）

面试官好，我是XXX，X年前端经验，核心技术栈**Vue3+TS+Vite/React+Next.js**，擅长小程序、H5开发和前端工程化落地。
过往做过电商、后台管理类项目，负责核心模块开发和问题排查，具备独立开发和团队协作能力，很希望能加入贵司，深耕前端技术，支撑业务发展，谢谢。

3. 自我介绍注意事项分析

- ❌ 禁止流水账：不说“毕业于XX、在XX公司做XX”，只说核心能力+项目成果+岗位匹配度；
- ❌ 禁止超时长：控制在1-2分钟，面试官没追问就不延伸，言简意赅；
- ❌ 禁止空泛：不说“我学习能力强、能吃苦”，用**技术栈+项目案例+落地结果**佐证；
- ✅ 突出重点：紧扣岗位JD，岗位要小程序就重点说小程序经验，要架构就说架构能力；
- ✅ 正向积极：全程无负面表述，不吐槽前公司/前项目；

- ✅ 收尾点睛：最后衔接对公司/岗位的认可，表达入职意愿。

4. 你为什么从上一家公司（上一份工作）离职呢？

✅ 优选正向理由（无任何抱怨，面试官最认可）

- ☒ 职业发展：前公司业务方向偏固化，贵司的【XX业务（如跨境电商/企业服务）】是我想深耕的领域，希望能接触更优质的业务场景，提升前端技术的落地深度；
- ☒ 平台提升：希望进入规模更规范、技术体系更完善的团队，学习更成熟的前端工程化、架构设计思路，实现个人技术和职业的双重提升；
- ☒ 地域原因：因个人生活规划（如定居深圳/家人在深），选择到深圳发展，匹配当地优质岗位；
- ☒ 业务转型：前公司业务重心调整，前端团队的技术方向与我的职业规划不符，希望寻找更贴合自身技术深耕方向的岗位。

❌ 绝对禁止理由：吐槽薪资低、领导不好、同事难相处、加班多、公司制度差，极易减分。

5. 请谈谈你的职业规划，你未来几年有什么目标？

短期（1-2年，落地型，贴合岗位）

深耕前端核心技术，快速吃透贵司的业务流程和技术体系，独立负责核心业务模块的开发、优化，解决项目中的技术难点，成为团队里能扛事的前端骨干；同时持续精进**TS/工程化/性能优化/跨端开发**能力，贴合业务落地更多技术方案。

中期（3-5年，提升型，有进阶性）

在前端技术深耕的基础上，向**前端架构师/技术负责人**方向发展，具备独立做技术选型、搭建项目架构、把控技术风险的能力，能带领小团队完成项目交付，同时结合业务场景探索前端新技术（如微前端、低代码）的落地，为团队提效、为业务赋能。

✅ 核心：规划贴合公司发展，不眼高手低，不空谈“创业/转行”，体现稳定性和成长性。

6. 在选择工作时，你最看重哪些因素？

按优先级排序（贴合职场心态，不功利，显诚意）

1. **业务与技术匹配度**：优先选择业务有前景、技术方向与我深耕领域（如前端架构/小程序）契合的岗位，能让技术落地产生价值，避免做重复无成长的工作；
2. **团队与成长空间**：希望团队技术氛围浓厚，有成熟的技术体系和分享机制，能跟着优秀的同事学习，获得技术和职业的提升机会，而非单纯执行任务；
3. **公司平台与稳定性**：认可公司的发展理念和行业口碑，平台规范、业务稳定，能让自己安心深耕，长期发展；
4. **合理的工作氛围与回报**：希望有健康的工作节奏，付出能得到合理的薪资回报和认可，兼顾工作效率和个人成长。

❌ 禁止只说“薪资高、加班少”，显得功利且无追求。

7. 你对加班有什么看法？

✅ 理性客观，体现责任心，不排斥不盲从

我认为加班需**分场景看待**：如果是项目紧急上线、突发技术故障、核心业务攻坚，我愿意主动加班配合团队完成任务，保障项目交付，这是作为技术人员的责任心；
但不认可无意义的低效加班（如磨洋工、流程冗余导致的加班），我更倾向于在工作时间内提升效率，通过**优化开发流程、复用组件、提前规避问题**减少不必要的加班；
同时也希望公司能平衡工作与生活，健康的节奏才能让团队更有持续战斗力，我也会在高效完成本职工作的基础上，配合团队的合理加班需求。

8. 在过去的工作中，你遇到过最大的挑战是什么？你是如何应对的？

前端岗经典案例（有挑战、有行动、有结果，STAR法则：场景-任务-行动-结果）

☒ **挑战场景**：负责的电商小程序项目，上线后出现**首屏加载超3秒、页面卡顿、支付环节偶发报错**的问题，用户投诉率高，领导要求一周内优化达标；

☒ **应对行动**：

1. ① 先定位问题：用小程序开发者工具分析性能瓶颈，发现是静态资源未压缩、接口请求串行、未做懒加载、支付回调逻辑有漏洞；
2. ② 分步解决：静态资源做gzip压缩+CDN分发，接口改为并行请求，非首屏组件/图片懒加载，优化支付回调的异常捕获和重试机制；
3. ③ 验证优化：上线灰度测试，监控加载速度和报错率，持续微调参数，同步输出性能优化文档，沉淀给团队复用；

☒ **最终结果**：首屏加载时间从3.2秒降至1.1秒，页面卡顿问题完全解决，支付报错率从0.8%降至0.02%，用户投诉清零，同时优化方案在团队其他项目落地，提升整体开发效率。

✅ 核心：挑战要具体（前端真实问题），行动要落地（技术手段），结果要量化，体现解决问题的能力。

9. 你之前是在广州工作，为什么选择来到了深圳？

✅ 正向理由，结合发展+个人规划，显稳定性

1. 1. **行业与机会**：深圳的互联网/科技产业更密集，前端技术的落地场景更丰富（如跨境电商、人工智能、企业服务），能接触到更前沿的技术和业务模式，对个人技术成长的帮助更大；
2. 2. **个人规划**：因个人/家人定居规划，决定长期在深圳发展，希望能找到适配的优质岗位，深耕技术，稳定发展；
3. 3. **平台优势**：深圳的企业技术体系更成熟，团队协作更规范，能让自己在更优质的平台上，实现技术能力和职业价值的提升，也更看好深圳的行业发展前景。

10. 你在团队合作中通常扮演什么角色？能举个例子吗？

前端岗核心角色（适配团队协作，不抢功，显能力）

✅ 核心角色：**技术执行者+问题解决者+协作配合者**，兼顾独立开发和团队协同，必要时承担技术攻坚和经验分享的职责。

案例佐证

在XX后台管理系统项目中，团队共5人（产品+后端+前端*2+测试），我主要负责**前端核心模块开发+技术问题兜底+联调协作**：

1. 1. 开发阶段：独立负责权限管理、数据可视化模块的开发，同时主动承接团队里的技术难点（如复杂表单校验、大屏适配）；
2. 2. 协作阶段：主动对接后端确认接口规范，提前规避联调风险；测试阶段配合测试排查bug，高效修复；
3. 3. 攻坚阶段：项目上线前发现数据渲染卡顿问题，我牵头分析原因，提出**虚拟列表+数据分片加载**的方案，快速落地解决，保障项目按时上线；
4. 4. 收尾阶段：整理项目开发文档、组件复用库，在团队内做技术分享，帮助新人快速上手。

最终项目顺利交付，团队反馈我的协作效率高，能扛事、能兜底，是团队里的核心前端支撑。

✅ 补充：可根据自身情况调整，比如偏“技术攻坚型”“细节把控型”，核心是有案例、有价值输出。

11. 你期望的薪资是多少？目前还有其他公司的Offer吗？

薪资回答（精准报价，有依据，不卑不亢）

结合我的X年前端经验、**Vue/React+TS+小程序/架构**核心能力，以及深圳同岗位的薪资水平，同时参考贵司的岗位定级，我的期望薪资是**XXk-XXk（月薪）**，年终奖希望能有X-3个月，整体年薪期望XXw-XXw。
这个薪资是基于我的技术能力、项目交付能力，以及能为公司创造的价值来定的，也希望能和公司的薪酬体系匹配，若双方契合度高，薪资可适当沟通协商。

Offer回答（真诚有度，显竞争力，不施压）

目前有1-2家同行业公司的Offer在沟通中，还未最终确定，因为我更看重贵司的**业务方向+技术团队+发展空间**，所以优先和贵司推进面试流程，希望能有机会加入团队，若贵司确定录用，我会尽快做决策。

❌ 禁止：报虚高薪资无依据、说“有很多Offer”施压、薪资一口价不协商。

12. 对于公司你有什么问题想问的吗？

✅ 必问问题（体现求职诚意，关注岗位/团队/发展，不踩雷，按优先级选3-5个）

🔹 岗位与工作相关（核心，体现想落地工作）

1. 1. 这个前端岗位的核心工作职责和考核指标是什么？主要对接哪些业务模块和团队？
2. 2. 岗位的技术栈是固定的吗？后续是否有机会接触微前端、低代码、跨端开发等新技术？
3. 3. 目前团队的前端项目处于什么阶段？是否有亟待解决的技术难点或优化方向？

🔹 团队与技术相关（体现关注成长，懂前端）

1. 1. 公司前端团队的规模、技术体系是怎样的？是否有技术分享、培训、晋升的机制？
2. 2. 团队的开发流程是怎样的？（如提测标准、上线流程、代码评审机制）
3. 3. 公司对前端技术的投入和规划是什么？是否支持前端技术创新和方案落地？

🔹 职业发展相关（体现稳定性，想长期发展）

1. 1. 这个岗位的晋升路径是怎样的？（如前端工程师→高级工程师→架构师）
2. 2. 入职后是否有导师带教？新人适应期的培训安排是怎样的？

❌ 禁止提问：薪资福利细节（已谈过）、加班时长、公司八卦、是否双休（可侧面问工作节奏）。

13. 你之前做过的项目都有哪些项目难点？你是如何解决这些问题的？

前端岗高频难点+解决方案（分场景，量化结果，可直接套用）

✅ 难点1：跨浏览器/跨端兼容性问题（PC端/H5）

☒ 问题：后台管理系统在IE11下出现样式错乱、部分JS功能失效，H5在微信/支付宝浏览器中支付跳转异常；
☒ 解决：① 样式层面：用CSS前缀兼容、重置样式库，规避IE不支持的CSS3属性；② JS层面：引入polyfill兼容ES6+语法，封装适配IE的工具函数；③ 跨端层面：做浏览器UA判断，针对不同浏览器写专属适配逻辑，支付环节对接各端官方SDK，做异常捕获；
☒ 结果：全浏览器兼容率100%，跨端支付成功率从98.5%提升至99.9%。

✅ 难点2：项目性能优化（小程序/PC/H5）

☒ 问题：小程序包体积超2M导致无法上线，PC端大数据渲染卡顿，H5首屏加载慢；
☒ 解决：① 小程序：分包加载、静态资源外链、删除无用代码/组件，包体积降至1.2M；② PC端：虚拟列表、数据懒加载、防抖节流，大数据渲染从5秒降至0.8秒；③ H5：资源预加载、接口并行请求、图片懒加载+压缩，首屏加载从2.8秒降至1秒内；
☒ 结果：项目顺利上线，用户体验大幅提升，性能指标达标，优化方案沉淀为团队规范。

✅ 难点3：复杂业务逻辑开发（如电商秒杀/风控校验）

☒ 问题：电商秒杀项目高并发场景下，出现库存超卖、下单重复提交、页面数据不同步；

☒ 解决：① 前端：做按钮防抖、提交状态锁、库存实时校验，避免重复请求；② 对接后端：采用接口幂等性设计、分布式锁，前端配合做请求重试和异常提示；③ 数据层面：实时拉取库存数据，做本地缓存+定时刷新，保障数据一致性；
☒ 结果：秒杀活动零超卖，下单成功率99.8%，无用户投诉，活动顺利落地。

✅ 难点4：前端架构搭建（新项目初始化）

☒ 问题：新项目需搭建高可用、易维护的前端架构，兼顾团队协作、工程化、可扩展性；
☒ 解决：① 技术选型：基于Vue3+TS+Vite搭建基础架构，集成ESLint+Prettier做代码规范，husky做提交校验；② 工程化：封装公共组件/接口/工具函数，实现路由懒加载、权限管理、环境区分；③ 扩展性：预留微前端、低代码集成入口，制定开发规范和文档；
☒ 结果：架构落地后，团队开发效率提升30%，代码质量显著提高，后续项目可快速复用架构，降低开发成本。

14. 抽述你参与设计或优化过的前端项目架构，包括关键决策和考虑的因素

前端架构设计（Vue3+TS为例，STAR法则，突出决策逻辑+落地价值）

✅ 项目背景：

主导XX企业服务平台前端架构搭建，团队5名前端，项目周期6个月，需支撑多模块、高并发、易维护的业务需求，同时兼顾新人上手效率。

✅ 关键架构决策：

1. 技术栈选型：Vue3 + TypeScript + Vite + Pinia + VueRouter4

☒ 决策考虑：① Vue3组合式API更适合复杂业务逻辑开发，TS保障类型安全，降低线上bug率；② Vite比Webpack打包速度提升80%，解决大型项目编译慢的问题；③ Pinia替代Vuex，简化状态管理，适配TS，团队学习成本低；④ 生态成熟，社区资源丰富，问题易排查。

2. 工程化体系搭建：规范+提效+质量把控

☒ 决策内容：① 代码规范：集成ESLint+Prettier+StyleLint，统一代码格式，husky+lint-staged做提交前校验，禁止不合规代码提交；② 模块化拆分：按「业务模块+公共模块」拆分，公共模块封装组件/接口/工具/指令，业务模块独立开发，降低耦合；③ 环境区分：配置开发/测试/预发/生产4套环境，隔离数据，避免生产环境污染；④ 构建优化：开启gzip压缩、按需加载、CDN外链静态资源，提升打包和加载效率。

3. 核心能力落地：权限+性能+可扩展性

☒ 决策内容：① 权限管理：基于「路由权限+接口权限+按钮权限」三级架构，结合后端返回的权限列表，动态生成路由，控制页面和按钮访问；② 性能优化：路由懒加载、组件按需引入、图片懒加载，

接口请求封装拦截器，做请求防抖/节流/重试；③ 可扩展性：预留微前端集成入口，支持子应用独立开发部署；封装业务模板，新人可快速复用开发。

4. 协作流程设计：提效+降风险

☒ 决策内容：① 代码管理：采用Git Flow分支管理，master为主分支，dev为开发分支，feature分支做功能开发，release分支提测，hotfix分支修bug；② 提测标准：制定前端提测 checklist，要求功能完成、兼容性达标、bug清零方可提测；③ 文档沉淀：编写架构文档、开发规范、接口文档、组件文档，团队共享，降低沟通成本。

✅ 决策考量核心因素：

1. 1. **业务适配**：架构必须贴合企业服务平台的复杂业务逻辑，支撑多模块并行开发；
2. 2. **团队适配**：兼顾团队技术水平，降低学习和维护成本，新人能快速上手；
3. 3. **质量保障**：通过TS、代码规范、权限管理，降低线上bug率，保障系统稳定性；
4. 4. **性能与效率**：优化打包、加载、开发效率，提升用户体验和团队交付效率；
5. 5. **可扩展性**：预留技术升级和业务拓展入口，避免后期重构成本过高。

✅ 架构落地结果：

项目顺利交付，线上bug率低于0.1%，团队开发效率提升40%，新人上手周期从2周缩短至3天；架构方案被公司其他前端项目复用，成为公司前端标准架构模板。

15. 项目初期，你如何制定技术选型和评估潜在的技术风险？

一、前端技术选型的核心流程（有理有据，不盲目选型）

✅ 第一步：明确选型前提（先定需求，再选技术）

1. 1. 梳理**业务核心需求**：是小程序/PC/H5/跨端项目？是否有高并发、大数据渲染、低代码、微前端等特殊需求？
2. 2. 确认**团队能力**：团队成员熟悉哪些技术栈？学习新技术的成本和周期是多少？
3. 3. 明确**项目指标**：项目周期、性能要求、兼容性要求、维护周期、预算成本。
4. 4. 参考**行业标准**：同业务场景的主流技术选型，避免选小众、无维护的技术。

✅ 第二步：技术选型的核心维度（前端核心模块，逐一评估）

选型模块	选型考量因素	主流选型（前端岗）
核心框架	业务复杂度、生态成熟度、TS支持、团队熟练度	Vue3/React+Next.js
构建工具		Vite/Webpack

	打包速度、生态、配置便捷性、性能优化能力	
状态管理	业务复杂度、TS支持、易用性、性能	Pinia/Zustand/Redux-Toolkit
路由管理	路由守卫、动态路由、嵌套路由、权限控制能力	VueRouter4/ReactRouter6
网络请求	拦截器、取消请求、重试机制、兼容性、易用性	Axios/Fetch+封装
UI组件库	组件丰富度、兼容性、定制化能力、体积、美观度	Element Plus/Ant Design Vue/Taro UI
工程化工具	代码规范、提交校验、打包优化、自动化部署	ESLint/Prettier/Husky/Rollup
跨端/小程序	跨端能力、原生支持、性能、生态	Taro/Uniapp/Nuxt.js

✅ 第三步：选型验证与决策

1. 1. 小范围**技术验证**：针对核心技术栈，搭建demo验证是否满足业务需求，测试性能、兼容性、易用性；
2. 2. 团队**评审沟通**：组织前端团队评审选型方案，收集意见，平衡技术优势和团队成本；
3. 3. 最终**选型定稿**：输出技术选型文档，明确各模块选型结果、选型理由、使用规范，存档备查。

二、潜在技术风险的评估与规避（前端核心风险，提前预判，制定预案）

✅ 风险评估的核心维度（前端高频风险）

1. 1. **技术选型风险**：选小众技术→生态差、问题难排查；选新技术→团队学习成本高、稳定性不足；选旧技术→性能差、无维护、后期升级难；
2. 2. **性能风险**：项目体积过大、加载慢、大数据渲染卡顿、高并发场景下请求异常；
3. 3. **兼容性风险**：跨浏览器/跨端/跨设备适配问题，功能失效、样式错乱；
4. 4. **团队协作风险**：代码规范不统一、分支管理混乱、接口对接不畅、提测标准模糊→效率低、bug多；
5. 5. **维护风险**：架构设计不合理、代码耦合度高、无文档沉淀→后期维护成本高、新人上手难；
6. 6. **线上稳定风险**：权限控制不足、接口异常无处理、数据校验缺失→线上bug、数据泄露、系统崩溃。

✅ 风险规避的核心措施（提前落地，降低风险）

1. 1. **技术选型风险规避**：优先选**成熟主流+团队熟悉**的技术栈，新技术小范围试点后再落地；避免技术堆砌，够用即可，预留技术升级入口；
2. 2. **性能风险规避**：初期制定性能指标（如首屏加载<1.5秒、包体积<2M），架构设计时融入性能优化方案（懒加载、按需加载、资源压缩）；
3. 3. **兼容性风险规避**：明确项目兼容标准（如浏览器IE11+/微信浏览器8.0+），选型时优先选支持兼容标准的技术，demo阶段充分测试兼容性；
4. 4. **团队协作风险规避**：制定代码规范、分支管理规范、接口对接规范、提测标准，集成工程化工具强制约束；建立每日站会、周评审机制，及时解决协作问题；
5. 5. **维护风险规避**：架构设计遵循**高内聚、低耦合**原则，模块化拆分；完善文档沉淀（架构、开发、接口、组件文档）；封装公共模块，避免重复代码；
6. 6. **线上稳定风险规避**：做完善的**异常捕获**（接口、JS、网络），制定重试机制；严格的**权限控制**和**数据校验**；上线前做灰度测试、压测；建立线上问题应急响应机制。

✅ 风险兜底方案：

输出**技术风险评估文档**，明确风险点、风险等级、影响范围、规避措施、应急方案；项目开发过程中定期复盘风险，及时调整优化，确保风险可控。

16. 描述一次团队合作中遇到的挑战及解决方案

前端团队协作经典案例（真实场景，体现沟通、解决、协作能力）

✅ 挑战场景：

XX跨境电商项目，团队由产品、后端*3、前端*2、测试*2组成，项目上线前2周，出现**前端与后端接口联调效率极低、需求频繁变更、测试bug堆积**的问题，项目进度滞后，团队氛围紧张，面临上线延期风险。

✅ 核心问题拆解：

1. 1. 接口问题：后端接口文档不规范、字段频繁变更、接口返回数据格式不一致，前端开发完后反复修改；
2. 2. 需求问题：产品在开发阶段频繁加需求、改需求，未走变更流程，前端返工量大；
3. 3. 协作问题：团队无每日同步机制，问题积压后才沟通，前端发现问题后无法及时对接后端/产品，效率低下；
4. 4. 测试问题：前端未自测就提测，bug多，测试反馈不及时，返工率高。

✅ 解决方案（主动牵头，落地执行，多方协同）

1. 解决接口联调问题（核心）

☒ 牵头组织**前后端接口评审会**，统一接口规范：明确接口请求/返回格式、字段类型、异常码规则，后端更新标准化接口文档（用Swagger），前端基于文档开发；
☒ 建立**接口联调群**，后端接口开发完成后，第一时间通知前端联调，实时同步问题，当天问题当天解决，避免积压；
☒ 封装前端**接口拦截器**，统一处理接口异常、数据格式转换，减少因后端数据格式问题导致的返工。

2. 解决需求变更问题

☒ 对接产品，制定**需求变更流程**：任何需求变更必须提交变更申请，评估对项目进度的影响，经团队评审通过后再执行，禁止无流程改需求；
☒ 梳理现有需求，区分**核心需求**和**非核心需求**：核心需求优先开发，保障上线；非核心需求延后至二期迭代，避免影响主流程。

3. 优化团队协作机制

☒ 推行**每日15分钟站会**：同步当日工作进度、遇到的问题、需要的支持，快速协调资源，解决卡点；
☒ 建立**问题台账**：记录项目中的问题、责任人、解决时间，实时更新，确保问题闭环；
☒ 明确**岗位职责**：前端负责前端开发+自测，后端负责接口开发+联调，测试负责提测后bug反馈，各司其职，减少推诿。

4. 降低测试返工率

☒ 制定**前端自测标准**：功能完成后，前端先自测功能、兼容性、异常场景，自测通过后再提交测试，附自测报告；
☒ 测试与前端**一对一对接**：测试发现bug后，直接对接对应前端开发，实时沟通bug原因，快速修复，提升bug解决效率。

✅ 最终结果：

1. 1. 接口联调效率提升60%，前端返工量减少70%；
2. 2. 需求变更得到有效管控，项目进度逐步追回；
3. 3. 团队协作氛围改善，问题解决效率大幅提升；
4. 4. 项目如期上线，线上bug率低于0.2%，获得领导和团队的认可；
5. 5. 制定的协作流程、接口规范被公司纳入团队协作标准，推广至其他项目。

17. 其他技术问题学好咱们的前端八股文

✅ 核心复习方向（前端面试必考，覆盖基础+进阶+工程化）

🔹 基础核心：

HTML5新特性、CSS3核心（flex/grid/动画/兼容性）、JS核心（原型/闭包/异步/Promise/ES6+）、DOM/BOM操作；

🔹 框架核心：

Vue3（组合式API/响应式原理/Pinia/Vite）、React（Hooks/虚拟DOM/状态管理/Next.js）、小程序（生命周期/分包/性能优化）；

🔹 工程化：

Webpack/Vite构建原理、ESLint/Prettier规范、Git分支管理、CI/CD自动化部署、前端模块化（ESM/CJS）；

🔹 性能优化：

前端性能指标、首屏加载优化、小程序/PC/H5性能优化、图片/资源优化、代码优化；

🔹 进阶技术：

TS核心（类型/泛型/接口）、微前端、跨端开发（Taro/Uniapp）、低代码、前端安全（XSS/CSRF/防劫持）、接口安全；

🔹 项目相关：

项目架构设计、技术选型、问题排查、解决方案、经验沉淀。

✅ 复习建议：按「基础→框架→工程化→性能→进阶」顺序，结合项目案例理解，而非死记硬背，能讲清原理+落地场景即可。

（注：文档部分内容可能由 AI 生成）