我的主要负责业务是当但部门活动和官网模块开发的B角

主要包括独立开发个游戏活动和官网前端页面，对接过的游戏有…

偶尔也会开发h5游戏的需求，如：。。。

除了前端页面开发，也会根据不同需求的变化，开发活动、小游戏后台接口，并且归纳总结接口文档提供给组内其他人员后续开发做参考

此外，还独立发展前端埋点工具的开发，，包括前端埋点上报插件和数据后台图形化查询统计分析界面

在前端页面开发方面：我能做到独立完成开发，包括需求分析、沟通协作、代码开发、自测、交付测试、更新上线等，比如：tnzmzx和rxlr预约页都是我自己一个人独立开发完成的

并且能做到根据需求的紧急程度合理划分规划事件、按时按质完成，比如：。。。

在完成需求的同时，也会追求页面的性能，如：。。

下面是一个开发需求案例

在需求分析中，发现需求方提出一个需通过登录mover平台，就直接通过手机号抽奖的需求，他们这样做的目的是：

1. 通过奖励去吸引用户预约，并且避免用户在不知情的情况下，用非绑定的手机号预约，导致无法发奖
2. 同时减少用户登录操作，减少用户的流失

但是不走登录抽奖的化，会存在以下问题：

1. 目前发奖接口是需要传用户的登录信息的，不允许直接通过手机号发奖
2. 如果实现一个只通过手机号发奖的接口的话，会有安全隐患，像：

如何兼顾减少用户操作并且抽奖

经过和需求方讨论，总结了两个方案

来的实现部分，这时就有两种实现方法：

1. 前端写死抽奖逻辑和抽奖结果，但是这样会出现。。。
2. 开发一个虚假抽奖接口，不做任何个发奖记录，这样的好处是模拟真实的抽奖过程，并且避免用户发现端倪

除了完成自己开发需求同时，还会帮助需求方仔细检查，想是 是否有素材遗漏、埋点需求是否有遗漏、配置是否有误，像是AB测试的配置，他们有时改了会忘记同步，我这边就有时不时去检查一下配置，看看他们是否有更改或者配置有误

除此之外，我还会协助市场同学与数据组沟通，捞取一些条件比较繁杂的数据，如：

做个小结：

在前端页面开发中，我能做到对新需求新问题积极沟通协作，讨论解决方案

实现代码时，能考虑不同实现方法的优缺点

测试时，仔细检查文档及相关配置问题，及时提醒需求方

除了前端页面开发，时常也会开发活动、游戏的接口，就比如之前开发的。。。

需求是微信小程序上实现羊了个羊的游戏玩法，并且计入抽奖、领奖、排行榜等功能，而且不走平台登录，所以基本上所有接口都得单独实现，并且绑定微信的openid

主要的难点在于防刷，参考了以往的游戏接口，防刷操作主要为以三步，

每局游戏生成一个gameid，并且任何游戏操作都要校验该id

然后开始构思游戏逻辑，由于游戏逻辑都靠后端校验，基本上就是后端在跑游戏，所以需要构思整个羊了个羊的逻辑，像是：。。。。

然后根据逻辑去设计每张拍的数据结果，包括id，状态、行列坐标、类型等

总结：主要学会了如果防刷，并且

埋点，事件追踪（Event Tracking），用于跟踪用户在页面上的行为和操作，并将相关数据发送到后端进行存储和分析。

**产品优化**：通过对用户行为数据的分析，可以了解用户的兴趣和需求，制定更加精准的运营策略，提高转化率和留存率，如：监控不同首批kv下，用户点击预约按钮次数，判断用户的喜好，提升预约转化率。

**页面性能监控**：收集页面性能指标数据，如：平均加载时间，及时对页面做性能优化，提升交互体验，减少用户流失

**其他优势**：

旧版ga也要停用，新版ga观察数据不直观

把埋点数据接入数据后台，保证数据安全性，方便查询具体数据，如查预约手机号等

**自研埋点工具：**

对比之前一直使用谷歌的第三方埋点分析工具，有以下优点：

1. 低数据延迟：ga数据收集需要经过24-48小时的数据处理和汇总，而数据后台则能够在较短的时间内（一两分钟）快速计算和呈现结果。
2. 精准投放：更快的改善页面效果，挖掘用户喜好，减少投放成本
3. 定制化程度高：根据业务需求，自定义上报数据内容，不受第三方工具影响限制，具有更高的灵活性和可扩展性，

如：

上包预约手机号，对不同渠道手机号预约数据分类汇总。

查询不同页面模块，用户的流失情况及预约情况

**研究GA埋点方案：**

一开始最大的问题是没有类似的项目经验，对埋点概念及其实现模糊不清，所以先从日常的活动埋点开始研究，了解活动的埋点方式及上报的埋点数据内容。之后阅读ga的官方文档，大致属性ga分析工具的使用及相关指标的概念和计算方式，并作出归纳总结

**开会讨论：**

开会讨论的目的是了解清楚市场运营所关注的数据指标，实现相关数据的收集上报

并向市场同学讲述ga分析工具的指标计算方式，便于他们更好理解。如：ga的许多指标都是基于会话实现的，讲解ga会话概念。

之后讨论并总结需要上报数据后台的数据指标。

**性能监控：**

参考ga文档，调用performance的相关api实现时间指标计算，由于performance.timing即将弃用，但getEntires api兼容性不是特别高，所以做了兼容出来，同时实现了两种方案去计算时间指标。

除此之外，还接入了web vitals库，计算并收集页面性能数据

**事件上报：**

主要解决将数据同时上报给ga和数据后台，并且要考虑旧版ga停用后，能将数据无缝切换上报给GA4，阅读了相关文档，发现可用gtag.js做旧版ga和GA4的数据上报兼容，于是将gtag.js接入埋点工具，在上报数据后台的同时，同时调用gtag.js的api上报ga。

**AB测试数据区分：**

活动的AB测试是用谷歌的优化工具实现，但是埋点工具无法识别优化工具对页面的更改，为了实现AB测试数据区分，了解了优化工具的运行逻辑，并通过优化工具的代码注入功能，注入一个区分不同页面效果的变量，通过识别该变量来实现数据区分

**前端埋点开发：**

研究前端的性能监控实现方案，如何计算加载、白屏、响应等事件指标

研究旧版ga和GA4的数据上报方式，并作两者的兼容，同时对接数据后台接口，最终将数据同时上报给三者

研究GA的会话计算方式、实现ga的转换率和跳出率计算

研究如果通过谷歌的优化工具达成对上报给数据后台的数据做AB测试区分

**为什么需要图形化界面：**  
查询的数据不够直观，数据后台总数据查询，会一次性查出不同维度下对应所有指标的数据，加上没有将数据图形化，导致数据量过大的同时，也无法很好的查看数据的变化趋势

缺少了对数据划分，比如：性能相关指标数据、事件指标数据和实时数据杂糅在一起，无法很好区分查询。

**图形化界面：**

性能监控页面：通过数据图形化，更好看出数据变化趋势，及时发现问题并改善页面问题

事件查询页面：可通过筛选特定的活动或事件，进行特定事件数据的搜索，并通过图形化展示数据趋势

实时数据页面：通过观察30分钟内用户的实时数据，可直观分析页面访问情况

**分析需求：**

参考ga，划分成性能、事件、实时页面。在埋点代码设计的时候就有考虑到区分不同的事件类型，通过上报event\_type来区分用户触发事件和非用户触发事件（如性能指标事件）

**设计原稿：**

一开始是考虑用mp设计原型的，但是了解现在比较火的在线设计软件figma的优势之后，决定使用figma来设计

**优化及交互：**

结合数据后台的设计风格，将原稿ui优化成初版ui，并增加新的数据筛选和分类功能。

后面在优化了ui颜色风格不统一问题，直接采用elementUI主题色。

完成页面开发后，总结了相应的操作文档，并且增加了数据后台总数据查询的操作文档。

之后收集用户的使用反馈，并且做交互和功能的优化。