**需求分析多介绍一些自己用到的框架等技术栈，吹起来**

**写的时候带好编号，星期三晚上12点之前交给盛亚堃整合**

**每一条需求用加粗四号黑体，对需求的分析用不加粗小四号黑体，后面有例子**

**需求和对需求的分析之间另起一行，前面空两格**

**算法端（1）支持边缘防护、明火烟雾检测、围栏翻越检测、挥手/抽烟/跌倒的行为检测、积水及渗漏水检测等校园安全相关场景；**

**算法端（2）对于围栏翻越检测、边缘防护等需要设定检测区域的场景支持良好的交互配置方式；**

**前端（3）在系统中对于所识别的事件在列表中进行展示，需要包含告警事件所产生的区域、时间、内容及所记录的摄像头；点击查看按钮可以查看事件的详细内容，包括视频及图片数据；支持按不同属性筛选事件；**

**前后端+算法端（4）支持对所识别到的违规事件进行告警，单次告警以事件为单位；支持推送信息至指定人员或公众号；支持优化报警频率；（毕师傅写识别到的违规事件进行告警，盛亚堃去写推公众号的介绍和接口的json格式，王子去写优化报警频率的实现）**

**前端（5）支持处置报警事件，未处置的报警高亮提示，已处置的报警可记录处置情况；**

**前后端（6）系统可接入模拟现场的测试视频流，可以完成测试操作并直观显示业务结果；（毕师傅写测试流的展示，盛亚堃去写rtsp转flv的实现）**

**后端（7）支持模型服务的二进制封装和灵活组合，可配置为高并发低时延、高资源利用率等多种模型服务方式适应不同项目需求；**

**后端（8）支持模型服务的二进制封装和灵活组合配置，适应不同项目需求，方便模型更新及功能扩展需求；**

**算法端（9）支持训练过程自动化、规范化，实现对不同模型的数据处理、模型训练、评测、量化等不同功能进行灵活组合和保存，支持快速且仅进行必要计算的重复训练，以支撑更多部署场景（如增加新的学校需补充新数据进行训练等）；**

**前后端（10）支持完善的监控和日志，便于观测系统的性能和资源使用；（毕师傅写前端监控和日志的展示，盛亚堃去写日志的获取实现）**

**一起写（11）系统具有良好的性能、功耗表现和可扩展性和成本优势，可支撑区县级数量规模的学校的接入；**

**算法端（12）通过人工智能支持对人员探出围栏的状态进行识别；**

**①围栏防护区域定义为，有围栏且围栏区域超过人员腰部区域；**

**②以人员身体腹部为准，腹部超出围栏内侧视为探出，并进行告警；**

**③计算探出以围栏外侧边为准线，超出围栏内侧边即视为探出。**

**算法端（13）通过人工智能提供对于人员待在危险区域时间过长进行识别；**

**①边缘防护区域指定为无围栏或围栏高度不超过成年人腰部的区域；**

**②对于边缘防护区域以人员出现在该区域时间进行告警；**

**③人员停留x秒后产生告警，x的具体值可配置。**

**算法端（14）通过人工智能提供对挥手、抽烟、跌倒行为进行识别；**

**①对人员挥手、抽烟、跌倒等行为进行识别；**

**②可以应用于人员密集区域，在人员密集区域保持识别效率。**

**算法端（15）通过人工智能提供对路面积水、墙面房顶渗漏水进行识别；**

**①可识别雨天路面积水情况，对于严重积水进行告警；**

**②可识别机房等房屋墙壁上的渗水情况并进行告警；**

**云端（16）基于K8s和微服务架构，提供良好的容器封装和高性能的接口调用，支持在K8s集群完成部署；**

**完整需求分析文档**

**（7）支持模型服务的二进制封装和灵活组合，可配置为高并发低时延、高资源利用率等多种模型服务方式适应不同项目需求：**

后端接收前端不同类型任务的灵活组合，并将任务需求添加到数据库中。使用goroutine和sync包中的WaitGroup来实现轻量级高并发，实现了模型服务的低时延，充分地提高了资源的利用率。遍历从数据库中检索到的所有摄像头记录，并为每个摄像头创建一个goroutine线程来调用编写的Judge函数进行处理。实现了并发处理多个摄像头的数据，提高了运行的效率。

使用了sync.WaitGroup的Add方法来增加计数器，表示要等待的goroutine数量。每个goroutine在处理完毕后都会调用WaitGroup的Done方法来减少计数器的值，表示已完成一个goroutine的处理。使用WaitGroup的Wait方法可以阻塞程序，直到计数器的值为0，即所有goroutine都已完成处理。

在每个goroutine线程内部，使用了defer语句来确保在函数执行结束后调用WaitGroup的Done方法，以便适当地更新计数器的值。

对于多个线程之间存在的共享内存竞争，使用了读写锁、互斥锁等来对多线程的共享数据来进行限制，实现了数据的一致性。

**（4）支持对所识别到的违规事件进行告警，单次告警以事件为单位；支持推送信息至指定人员或公众号；支持优化报警频率：**

后端使用gin-web框架通过处理HTTP请求，实现了推送信息至公众号。首先，To\_weixin函数接收前端处理的POST请求，调用微信API向微信公众号发送消息。然后，调用Get\_token和Get\_openid函数通过发送Get请求，请求API来获得访问令牌和关注公众号的用户ID列表。最后，解析前端传来的JSON，并将其解析为Weixin\_Results结构，构造消息发送到WeChat组，使用WeChat官方API将消息发送给所有关注公众号的用户。使用此方法，降低了服务器压力，充分地提高了资源利用率，实现了低时延报警推送公众号。

**（6）系统可接入模拟现场的测试视频流，可以完成测试操作并直观显示业务结果：**

后端使用Go语言的Gin框架和GStreamer，通过gst-launch-1.0工具，实现了从接入模拟现场的测试RTSP视频流中提取H.264编码的视频数据，将其解码后，使用flvmux元素封装为FLV格式的视频文件，并将其保存在磁盘内，并将生成的FLV文件作为HTTP响应的内容类型，设置HTTP响应头为“video/x-flv”，将其传输给客户端。通过使用FFmpeg和GStreamer这两个多媒体框架，低资源占用率的条件下实现了视频的解码、编码、转换等操作；在同一个函数中并发地实现了客户选中的8个重点地区的rtsp视频流到flv格式文件的转换，实现了业务的灵活组合配置，可以适应不同项目的需求。

**（8）支持模型服务的二进制封装和灵活组合配置，适应不同项目需求，方便模型更新及功能扩展需求；**

使用Docker这一支持二进制封装和灵活配置的框架，可以将封装好的模型镜像文件部署在不同的环境中，通过将模型封装为镜像文件并进行灵活组合配置，实现了可以方便地进行模型更新和功能拓展需求，具有更好的可移植性和拓展性。

使用Kubernetes进行模型服务的管理和部署，其提供的可拓展容器编排平台，实现了自动化部署、拓展和管理容器应用，实现了均衡负载的功能。

**（10）支持完善的监控和日志，便于观测系统的性能和资源使用：**

后端保存了每一条算法端实时返回的检测数据，低延时地添加到数据库中，作为返回给前端的日志内容。将检测数据添加到数据库使用原生SQL语句，使项目具有更高的灵活性、更好的可控性和更好的可移植性，可以更方便地移植到其它的数据库系统中，更简单地控制查询的执行计划和优化，确保了SQL语句的执行效率和性能，实现了检测数据实时地插入日志。

后端使用了Go语言的Gin框架和runtime包，实现了快速、高性能的CPU监测功能与获取当前程序的内存使用情况功能，将其以JSON格式返回给客户端。通过runtime包中的函数获取当前的CPU核心数与对内存使用情况的监测，并使用time包中的函数计算CPU使用情况。然后代码将结果封装成CPUUsage与MemUsage结构体，将其转换为JSON格式返回给前端。这对于检测系统的性能和资源使用非常重要，充分地体现了使用Go语言原生的内存管理机制的优势，可以帮助开发人员有效地避免常见的内存错误，便于观测系统的性能和资源使用情况。