# 软件设计规格说明书（SDS）

## 1. 引言

### 1.1 目的

本项目的目的是通过计算机视觉技术，特别是深度学习模型（例如 YOLOv5），实现对玻璃内部爆炸的自动检测。通过对大厦和其他玻璃富集区域进行实时监测，我们希望能够及时发现并识别内部爆炸现象，为建筑管理者提供一个强有力的工具，以便他们可以采取及时的修缮措施，确保建筑结构的安全和稳定。

### 1.2 问题陈述

内爆问题可能由于多种因素引起，包括结构紧张、温度变化、制造缺陷等。传统的检测方法可能面临效率低、耗时长的问题。通过引入自动化的计算机视觉检测系统，我们的目标是提高内爆检测的速度、准确性和实时性，使管理者能够更加迅速地做出响应。

### 1.3 项目重要性

实现一个可靠的玻璃内部爆炸检测系统对于建筑管理和维护领域具有重要意义。它不仅有助于提高建筑结构的安全性，还可以减少因内爆问题引起的损害和维修成本。通过及时检测和修复，我们可以有效地维护建筑结构的稳定性，确保大厦和其他玻璃结构的安全运行。

通过本项目的实施，我们期望为建筑管理领域引入先进的技术手段，提升建筑结构的安全性和可维护性。

### 1.4 范围

该软件设计规格说明书适用于玻璃内爆检测的设计阶段，详细描述了系统的结构和各个组件之间的关系。

## 2. 系统架构设计

### 2.1 系统概述

（简要描述整个系统的概况，包括系统的主要功能和特性）

### 2.2 系统结构

（提供系统的高层次结构图，标识主要组件和它们之间的关系）

### 2.3 系统模块

（列出系统的各个模块，并为每个模块提供详细描述）

## 3. 模块设计

### 3.1 模块1

#### 3.1.1 模块概述

（描述模块的主要功能和目标）

#### 3.1.2 模块接口

（定义模块与其他模块之间的接口）

#### 3.1.3 数据流

（描述模块之间的数据流，包括输入和输出）

### 3.2 模块2

（按照相同的模式描述其他模块）

## 4. 数据库设计

### 4.1 数据库概述

（描述系统使用的数据库，包括数据库类型、表结构等）

### 4.2 表设计

（为每个数据库表提供详细描述，包括字段、数据类型、约束等）

## 5. 用户界面设计

1. 粒子效果部分

使用了 vue-particles 插件来创建动态的粒子效果背景。

:particlesInit 和 :particlesLoaded 是作为插件的初始化和加载事件的回调函数，通过它们可以执行一些自定义的操作。

配置项中包含了粒子的颜色、数量、运动方式等参数，可以根据需要进行调整。

1. 文件上传界面

.contain1 区域包含一个文件选择输入框，通过点击选择文件。

用户通过选择本地文件触发 handleFileChange 方法。

1. 图片预览和检测结果显示

.building 类用于包装图片预览和检测结果展示，设置了样式包括背景颜色、边框等。

图片预览区域包含一个动态显示用户选择图片的 img 元素，通过 previewUrl 实时更新。

检测结果区域包含一个动态显示检测结果图片的 img 元素，通过 imagePath 实时更新。

1. 按钮交互

.mybutton 类用于包装按钮，设置了按钮的样式。

使用了 TweenMax 和 GSAP 库实现了按钮点击时的动画效果。

点击按钮触发 handleButtonClick2 方法，该方法通过 axios 发送 POST 请求将选择的图片上传到服务器，并接收处理结果。

1. 上传图片和处理

handleFileChange 方法监听文件选择输入框的变化，获取用户选择的文件。

uploadImage 方法创建一个 FormData 对象，将用户选择的文件附加到其中，然后通过 axios.post 发送 POST 请求到服务器。

服务器通过 /upload 接口处理文件上传，保存文件并调用 mymain 函数进行检测，返回检测结果的图片的 Base64 编码。

1. Vue.js 组件和生命周期

使用了 <template>, <script setup>, <style> 标签定义 Vue.js 组件。

使用 onMounted 钩子在组件加载后执行一些初始化操作。

1. 动画效果

使用了 GSAP 库的 TweenMax 功能，实现了按钮点击时的动画效果，包括缩放、弹性等。

1. 整体布局和样式

使用 Flex 布局使得界面能够在不同屏幕尺寸下灵活显示。

设置了元素的样式，包括背景颜色、边框、字体等，通过样式创建了整体的美观效果。

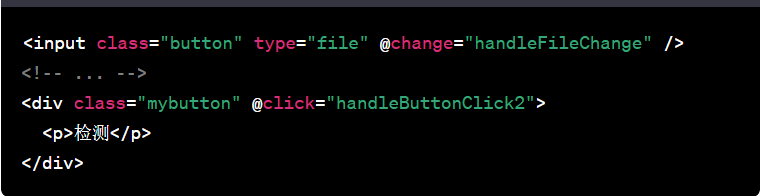
## 6. 接口设计

### 6.1 外部接口

（描述系统与外部系统或硬件之间的接口）

### 6.2 API接口

1. 文件上传部分



用户通过点击"选择文件"按钮来触发文件选择操作。

选择的文件将通过handleFileChange方法进行处理。

当用户点击"检测"按钮时，将调用handleButtonClick2方法，该方法会调用uploadImage函数。

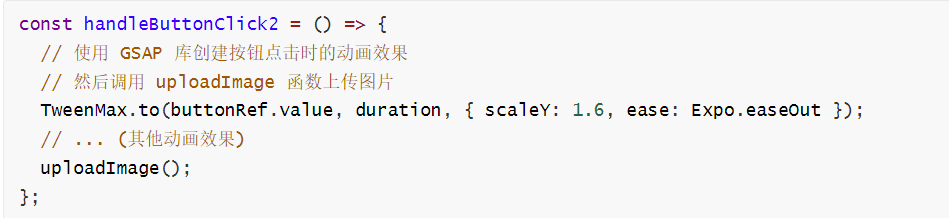
1. 文件处理逻辑



uploadImage函数使用axios库发送POST请求，将选择的文件上传到服务器的/upload接口。

服务器返回处理后的图片数据，将其显示在检测结果区域。

1. 动画效果



使用 GSAP 库创建按钮点击时的动画效果，例如缩放效果。

动画完成后调用uploadImage函数上传图片。

1. /upload 接口：通过 POST 请求接收文件上传，进行处理后返回检测结果的图片。该接口对应的处理函数 upload\_file 接收上传的文件，保存在服务器上，然后调用 mymain 函数进行检测，并将结果以Base64编码的形式返回。

文件上传接口：http://localhost:7078/upload

方法：POST

参数：使用 file 字段上传文件

处理函数：upload\_file

1. /download/<filename> 接口：通过 GET 请求提供文件下载服务，这个接口允许客户端根据文件名下载已经处理过的图片。该接口对应的处理函数 download\_file 从指定目录 (UPLOAD\_FOLDER) 中提供文件下载。

文件下载接口：http://localhost:7078/download/filename

方法：GET

参数：filename 是要下载的文件名，可以替换成实际的文件名

处理函数：download\_file

## 7. 安全性设计

### 7.1 访问控制

（描述系统中的访问控制措施）

### 7.2 数据加密

（如有必要，提供对敏感数据的加密设计）

## 8. 性能设计

### 8.1 性能目标

（定义系统的性能目标，如响应时间、吞吐量等）

### 8.2 优化措施

（描述用于优化系统性能的措施）

## 9. 测试策略

### 9.1 单元测试

（描述单元测试计划和方法）

### 9.2 集成测试

（描述集成测试计划和方法）

### 9.3 系统测试

（描述系统测试计划和方法）

## 10. 部署计划

### 10.1 硬件需求

（描述系统部署所需的硬件配置）

### 10.2 软件需求

（描述系统部署所需的软件环境）

### 10.3 部署流程

（提供系统部署的步骤和流程）

## 11. 可维护性和扩展性

### 11.1 代码结构

（描述代码的组织结构和模块化程度）

### 11.2 文档

（定义系统文档的标准和规范）

### 11.3 扩展性设计

（描述系统的扩展性设计和可维护性）

## 12. 风险管理

### 12.1 风险识别

（识别可能影响系统设计和实现的风险）

### 12.2 风险应对策略

（定义应对每个风险的具体策略和计划）

## 13. 参考文献

（列出在设计过程中参考的所有文献和资源）

## 14. 附录

（包括任何附加信息、图表、表格等）

---

请注意，上述只是一个通用的模板，实际的SDS内容可能会根据项目的特定需求而有所调整。在编写SDS时，确保团队成员之间的有效沟通，以确保设计文档涵盖了所有必要的方面。