**3.2 软件方案**

**3.2.1 设计思想**

分层设计：区分持久、业务层、原子业务层、展示层，保证层次清楚、低耦合，各层通过接口接入并进行参数校验，保证数据操作的安全

访问验证：系统进行用户身份权限验证，防止用户访问未授权页面

**3.2.2 总体目标**

要求保证系统网络和数据传输的安全

要求系统有较强的抗干扰性能，系统的局部故障不影响整体的运行

要求采用成熟和先进的技术

要求系统设计要充分考虑扩充的能力

**3.3.3 设计原则**

本项目的设计以需求为导向，尽量收集用户的要求，采用构件的设计思想，便于系统功能的重组和扩充。系统采用“总体规划、分步实施”的策略，第一步完成通用控制页面的设计，接下来完成常用的后台服务程序的开发，最终达到实现质量体系的目标。为保障项目的各项性能，本项目在设计和实施过程中应遵循如下的原则：

先进性：采用先进、成熟的信息技术，既要保证系统在满足当前的功能需求，也为将来的功能需求提供条件

实用性：充分考虑到质量管理和流程控制的实际需要，设计方案留有一定的余地，即在设计中构造明晰的体系结构，便于更改

可扩充性：采用面向对象的设计思想和开发技术以及构件设计思想，保证在用户的需求发生变化和增加时，使系统的修改量减到最小。

开放性：由于本系统涉及许多硬件设备和外部应用软件，为了更好地和各种硬件设备和应用软件接口，在设计时，应提供主要的接口标准。

标准化：在系统总体规划设计中，所有软、硬件产品的选择必须选择符合开放性和国际标准化的产品和技术；在应用软件开发中，必须遵循总体组制定的各项规范和要求。

安全与保密：系统设计时应充分考虑系统的安全，提供备份的功能，保证当系统出现硬件或软件的故障时，能进行恢复；另外，应对用户进行权限管理，保证信息的安全。

以数据为中心：本项目采用以主题数据库为中心的设计思想，首先把本项目的主题数据库设计出来，再在此基础上开发各种应用。由于主题数据库是项目的基本信息资源，它具有很好的不变性和稳定性，相对独立与具体的应用，当用户的需求发生变化时，使修改量减到最小。

**3.3.4 设计思想**

本项目采用通用的开放操作系统和开发平台，确保该软件的通用性和可扩充性。

考虑最小修改的原则，软件方案总体分为六部分，分别如下：

1、服务发布层：发布对应的系统服务供Web、微信、设备调用

2、业务组装及业务处理层：数据校验、日志记录、业务逻辑处理等

3、原子业务层：最小逻辑处理单位、持久层组装等

4、数据持久层：数据存储、查询、修改、删除

5、公共组件层：文件上传下载和读取、时间处理组件、函数处理组件等

6、配置层：程序配置文件等

**3.3.5 系统架构**

分层进行业务数据处理，相互独立，原则上同级不可调用，下级不可调用上级，不可越级调用

**3.3.6 统一的后台数据平台**

共用一个数据后台，保证数据的可靠性和唯一性。

**3.3.7 通过Json语言实现数据交换**

采用Json语言实现数据传递是主要考虑到，传输体积较小，占用内存空间较小，解析较快，与对象的转换较方便

**3.3.8 前端**

VUE 2.0、element ui，ios，android

可选择性添加模块

易于被开发者接纳

易于与其它库和工程集成

可以通过两种数据绑定方式更新模型和视图

适合于大型的应用

**3.3.9 后端**

**Php: 5.6.x**

**PHP框架: Codeigniter 3.1.9**

**3.4.1 部署思想**

统一由负载均衡器（Nginx）进行业务分发和路由，数据直接交由数据库（MySQL）