1. 提纲式和结构图式WBS
2. 提纲式

1 疫情信息手机系统

1.1项目规划

1.1.1讨论系统主要功能

1.1.2任务计划的编制

1.1.3任务计划的确定

1.2需求分析

1.2.1简约用户界面

1.2.2信息实时发布

1.2.3信息分类

1.2.4信息进度状况展示

1.2.5用户反馈

1.2.6权限管理

1.2.7安全可靠

1.2.8系统支持高并发操作

1.2.9使用微信小程序

1.3总体设计

1.3.1用户管理

1.3.2管理员

1.3.3信息发布

1.3.4信息展示

1.3.5信息分类

1.3.6用户反馈

1.3.7信息管理

1.3.8系统管理

1.3.9测试

1.4详细设计

1.4.1接口设计

1.4.2数据库设计

1.4.3用户登录模块设计

1.4.4信息管理模块设计

1.4.5系统后台管理模块设计

1.4.6社区端口模块设计

1.4.7信息优先度模块设计

1.4.8信息完成进度模块设计

1.5实现

1.5.1代码编写

1.5.2页面设计

1.5.3单元测试

1.6测试

1.6.1需求测试

1.6.2系统测试

1.6.3测试记录

1.6.4测试总结

1.6.5使用手册编写

1.7交付

1.7.1验收测试

1.7.2产品提交

1. 结构图式

图示

描述已自动生成

1. WBS字典

|  |  |
| --- | --- |
| WBS表示号 | 1.3.1 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 用户管理 |
| 描述 | 1. 储存用户信息数据：记录相应数据 2. 设置用户信息滤波：对某些信息可设置滤波 3. 浏览个人信息数据 4. 更改个人信息数据：为用户对自己的信息提供修改功能 5. 生成历史数据：可生成历史事件报告 |
| 完成的任务 | 1,2,3,4已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.2 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 管理员管理 |
| 描述 | 1. 储存管理员信息数据：记录相应数据 2. 储存事件数据：记录相应事件 3. 设置事件滤波：对某些事件可设置滤波 4. 浏览、更改事件日志：对所有事件提供浏览和更改状态功能 5. 设置权限信息：普通用户和不同社区端拥有不同权限 6. 管理用户信息：对不实言论进行处理 7. 生成历史数据：可生成历史事件报告 |
| 完成的任务 | 1,2,3,4,5已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.3 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息发布 |
| 描述 | 1. 储存信息数据：根据用户选择的关键词的不同向数据库记录用户发布的事件信息 2. 设置信息事件滤波：对无关信息设置滤波 3. 浏览事件日志：管理员对所有用户已发布的事件提供浏览功能，用户对自己已发布的事件提供浏览功能 4. 生成历史数据：可生成历史事件报告 |
| 完成的任务 | 1,2,3已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.4 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息展示 |
| 描述 | 1.信息展示及推送：对已分类好的信息根据关键词优先级的不同进行首页推送；设置不同关键词的信息入口；为普通用户和社区端展示不同页面 |
| 完成的任务 | 1已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.5 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息分类 |
| 描述 | 1.信息分类：后台通过用户所选关键词不同，将信息分类处理；根据不同的关键词，使用不同的接口从数据库筛选相应信息，展示于相应社区人员界面或普通用户界面。 |
| 完成的任务 | 1已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.6 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 用户反馈 |
| 描述 | 1. 用户所发布的信息，在得到解决后可进行反馈将其修改为已解决时间，同样长期未得到解决的信息也可进行反馈催促，系统收到后会进行处理。 |
| 完成的任务 | 已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.7 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息管理 |
| 描述 | 1. 信息浏览：设置浏览日志，管理员对信息浏览进行管理并生成历史数据。 2. 信息收集：通过个人发布、社区上报、政府发布渠道收集 3. 信息汇总、存储：：将所收集到的大量信息存储到数据库。 4. 信息分类、发布：根据不同的关键词，使用不同的接口从数据库筛选相应信息，展示于相关社区人员界面或普通人员界面。 5. 信息展示：根据关键词，向个人、社区展示相关信息。此处信息展示有不同的优先级管理及响应，如收到特别紧急事件，后台会发出弹窗提醒。 6. 信息处理：后台定时统计信息，每两小时（时间可根据疫情紧急状况进行相应调整）统计所有物资类信息，统一传递给下一部门分配，并将其显示在相应社区的信息展示页面下。（对紧急事件有不同的优先级响应） 7. 信息进度管理：分为已解决、为解决、正在解决。对于用户所发布的信息，在得到解决后可进行反馈将其修改为已解决时间，同样长期未得到解决的信息也可进行反馈催促，系统收到后会进行处理。 8. 信息保密：保证用户实名认证信息的安全性。 9. 设置信息事件滤波：对无关信息设置滤波 |
| 完成的任务 | 3、4、5、6、7、8、9、已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.8 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 系统管理 |
| 描述 | 1. 权限管理：根据不同的关键词，使用不同的接口从数据库筛选相应信息，展示于相关社区人员界面或普通人员界面。 2. 安全可靠：用户只能对自己的用户名、密码进行修改，实名信息一经认证无法修改，发布信息时仅显示所发布人的用户名；根据授权等级的不同，社区、后台管理员可访问不同级别的用户数据。 3. 系统易维护、易操作：系统信号调度灵活、科学。交互界面简洁明了、易学易用。系统总体布局合理、有效，不对硬件组态进行任何改变，系统硬件设备的链接直观简单、便于维护。 4. 系统支持高并发操作：高并发是互联网分布式系统架构设计中必须考虑的因素之一，保证系统能够同时并行处理很多请求。 |
| 完成的任务 | 1、2、3、4、已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.3.9 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 测试 |
| 描述 | 1. 功能测试：以需求文档和视觉交互文档为准。 2. 兼容性测试：操作系统的兼容性、屏幕的兼容性、微信的兼容性（版本）。 3. 网络测试：网络状态和环境的切换，断网，通过设置代理进行弱网的测试等等。主要是考察小程序在各种网络状况下的运行情况。 4. 接口测试。 5. 功能交互：因为小程序是在微信里面，所以还需要验证一些跟微信的交互。 6. 缓存：需要验证清空微信的缓存是否对小程序有影响。 7. 性能测试：页面的白屏时间，首屏时间，资源占用，页面渲染时间，帧率等等。 8. 渗透测试：在测试阶段发现高风险漏洞，提早进行修复，尽早避免因代码漏洞造成的安全风险损失。 9. 安全检测：覆盖前台代码安全和API使用规范，以及业务CGI和对WEB框架和的安全检测。 10. 小程序加固：针对小程序前端代码的加密服务。 11. 权限：访问权限是否授权。所以权限测试分为“已授权”和“未授权”，所以需要测试在跳转到微信小程序时“允许访问”和“不允许访问”这两种情况下小程序是否各项功能能够正常工作。 |
| 完成的任务 | 尚未完成 |
| 责任者 | 软件测试人员 |
| 完成的标识 | 通过软件测试部的质量测试报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.1 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 接口设计 |
| 描述 | 小程序数据存储处理  WHC.SmallApp.Data调用WHC.SmallApp.API进行处理。  1.调用公共实体类项目接口（WHC.Common.Entity）  2..Net公共类库  3.通用基础模块（WHC.Freamework.Data）  1)附件管理（Attachment）  2)字典管理(Dictionary)  3)权限管理模块（Security）  4)其他模块（Other）  4.主要接口  1）开放接口  2）微信登录（login）  wx.login(object);  3）获取用户信息（getUserInfo）  @click="getuserinfo;  wx.getUserProfile;  4)后台数据获取  data: { schooList: [], },  wx.request({}), |
| 完成的任务 | 尚未完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 设计出完整的接口 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.2 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 数据库设计 |
| 描述 | 1.需求分析  需求收集和分析，得到数据字典和数据流图。  2.概念设计阶段  对用户需求综合、归纳与抽象，形成概念模型，用E-R图表示。  3.逻辑结构设计阶段   1. 将概念结构转换为某个DBMS所支持的数据模型。 2. 每一个实体集对应一个关系模型，实体名作为关系名，实体属性对于相应的关系属性。   4.数据库物理设计  为逻辑数据模型选取一个最适合应用环境的物理结构。   1. 数据库实施   建立mysql数据库用于存储用户信息，建立MangoDB数据库，用于存储其余信息，因为其自带的日志功能，便于日常的维护与运营，编制与调试应用程序，组织数据入库，程序试运行，并将实际数据库资源部署在云数据库上，使其运行更高效、便捷。   1. 数据库运行和维护阶段   对数据库系统进行评价、调整与修改 |
| 完成的任务 | 尚未完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 设计出通过压力测试的数据库 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.3 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 用户登录模块设计 |
| 描述 | 1.角色：登录的用户，默认情况有三个不同角色，分别为：系统管理员，社区端，用户。  2.权限分配：用户登录之后按角色分配权限信息，社区端与用户是两个不同的界面  3.存储：登录日志表自动保存登录信息，权限信息保存在Session中  4.登录检测： Session中保存用户id和最后登录时间，成功后member表中更新最后一次登录的日期时间  5.个人信息修改：用户登录后可进行密码修改，用户登录后才能够注销账号。 |
| 完成的任务 | 1、2、3、4、5已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.4 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息管理模块 |
| 描述 | 1. 存储信息:根据用户所选择的分类将上传的信息存储入数据库中 2. 信息展示：将所发布的信息在用户与社区的界面同步展示 3. 信息管理：根据不同的分类将信息送入下一步进行处理，并为其设置进度标志，以确保为用户反馈此条信息处理状况 4. 信息查封：对于已经查证不实或恶意发布与本系统无关的信息，用户举报或后台搜索到后管理员可对此条信息进行删除操作，并对此用户实行限制 |
| 完成的任务 | 1、2、3、4已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 通过质量保证部的验收报告 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.5 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 系统后台管理模块设计 |
| 描述 | 1. 后台   后台维护用户数据，并用一个管理模块来对后台的权限、角色、密码和模块进行管理。  2、模块管理  对整个系统的设置和管理，少数管理员拥有管理权限。  3、权限系统  权限系统保证系统分工。在用户访问系统上时，系统按照权限系统的设置来控制用户的访问功能和数据。  4、角色管理  新增用户账号时，给用户分配角色。每个用户可以分配一个角色，用户可上传自己的信息。  用户管理员  可对用户的信息进行管理，对发布恶意造谣信息的用户进行拉黑或删除其账号  5、信息移交  对用户所发布物资紧缺类信息定时进行统计上报，并将信息移交给下一部门处理（若在此之前社区端已阅读此信息，并正在完成，此类信息不重复移交）  6、紧急状况提醒  若后台收到紧急状况类信息的发布，一经收到，立即通知社区人员入户查证并通知医院此情况 |
| 完成的任务 | 1、2、3、4、5、6已完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 设计出完整的后台管理模块 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.6 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 社区端口设计 |
| 描述 | 1. 社区登录：根据管理员所分配的账户登录 2. 更改个人信息：用户可更改密码 3. 提取社区相关数据：根据社区不同从数据库中找到所需的数据 4. 提交进度数据：在社区用户浏览信息后返回此条信息的进度状况 5. 存储数据：将社区用户所处理事件进度存储至数据库中 6. 生成历史数据 |
| 完成的任务 | 1、2、3、4已完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 设计出完整的社区端口 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.7 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息优先度模块设计 |
| 描述 | 1. 优先度分类：根据信息的大致情况将用户所发布的信息分为三类：物资紧缺类、紧急状况类以及普通病症类 2. 信息处理：根据不同类别的信息，后续采取不同的操作，如物资紧缺类与普通病症类后台定时进行统计移交下一部门；紧急状况类在收到信息时就通知社区进行查证，并告知医院此情况 3. 用户反馈：根据不同类别的信息，所设置的用户反馈时间也有所不同，由于紧急状况类信息须立即处理，故与用户进行反馈是否处理的时间也会随之加快 4. 存储数据：在信息已经处理后对于数据库中的信息进行修改并重新存储 5. 生成历史数据 |
| 完成的任务 | 1、2、3已经完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 信息完成优先度排序处理 |
| 备注 |  |

|  |  |
| --- | --- |
| WBS表示号 | 1.4.2 |
| 名称 | 疫情信息收集系统 |
| 主题目标 | 信息完成进度模块设计 |
| 描述 | 1. 进度更改：用户发布后的信息在经过每一次操作时，会对其进度属性进行及时修改 2. 进度展示：用户发布信息后，界面上可以看到所发信息的进度状况 3. 用户反馈：若进度长时间卡在一个地方未发生改变，用户可以进行反馈，以便于后台人员处理 4. 存储数据：更改后的进度属性重新存储入数据库中 5. 生成历史数据 |
| 完成的任务 | 1、2、3已完成 |
| 责任者 | 软件开发人员 |
| 完成的标识 | 节点通过测试，可以完成不同时间的反馈 |
| 备注 |  |

1. 敏捷开发的用户故事

As a user,  
I want to be able to view and publish information  
so that I can master information and express my views.  
  
As a user,  
I want to the interface of the applet is more concise  
so that  make the operation more concise.  
  
As a administrator,  
I want to information can be sorted according to emergencies  
so that I can prioritize more urgent events.  
  
As a administrator,  
I want to login with realname  
so that I can know the authenticity of information and facilitate management.  
  
As a busy execytive,  
I want to view project progress  
so that I can know how many projects have been completed.  
  
As a CTO,  
I want to small programs can be highly concurrent and easy to maintain  
so that reduce the cost of later maintenance.  
  
As a CTO,  
I want to the operating environment can be compatible with a variety of systems  
so that make sure more users can use it.

As a user,  
I want to see information progress status  
so that I know if my needs have been addressed.  
  
As a administrator,  
I want to users can feed back the information  
so that we understand users and get improved