**员工管理系统项目**

**(软件设计说明书)**

内部资料 注意保密

修订记录：

*此修订记录中的文字为模板信息，请在使用时注意按照实际情况填写。*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 修订人 | 审批人 | 日期 | 修订描述 |
| 1.0 | 何梓浩 | 冯振熙 | 1.8 | 1. 填写1与2章 |
| 1.1 | 冯振熙 | 何梓浩 | 1.9 | 1. 补全剩下内容 |
| 1.2 | 冯振熙 | 何梓浩 | 1.11 | 1. 修改错误内容 |
|  |  |  |  |  |

目 录

[1 文档介绍 4](#_Toc41401700)

[1.1 文档目的与范围 4](#_Toc41401701)

[1.2 读者对象 4](#_Toc41401702)

[1.3 参考文献 4](#_Toc41401703)

[1.4 术语与缩写解释 5](#_Toc41401704)

[2 假设前提 5](#_Toc41401705)

[3 概要设计（总体设计） 6](#_Toc41401706)

[3.1 模块功能列表(功能架构) 6](#_Toc41401707)

[3.2 模块间交互图 6](#_Toc41401708)

[3.3 模块重用设计【可选】 6](#_Toc41401709)

[4 数据库设计说明 8](#_Toc41401710)

[4.1 数据表规划 8](#_Toc41401711)

[4.2 ｛数据表1名称｝设计说明 8](#_Toc41401712)

[4.3 ｛数据表2名称｝设计说明 9](#_Toc41401713)

[5 模块1设计说明：模块名 9](#_Toc41401714)

[5.1 模块功能描述 9](#_Toc41401715)

[5.2 模块类图描述 9](#_Toc41401716)

[5.3 关键方法和过程 9](#_Toc41401717)

[5.3.1 关键方法列表 9](#_Toc41401718)

[5.3.2 方法1：方法名 10](#_Toc41401719)

[5.3.3 方法n：方法名 10](#_Toc41401720)

[6 模块2设计说明：模块名 11](#_Toc41401721)

[7 风险 11](#_Toc41401722)

[8 附录 11](#_Toc41401723)

# 文档介绍

## 文档目的与范围

本文档可以作为产品概要设计、详细设计以及测试的参考资料，方便各开发人员看懂项目

主要介绍模块、数据库以及函数

## 读者对象

建议以下人员阅读本文档：

1. 软件测试人员
2. 程序开发人员
3. 项目管理人员

## 参考文献

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编号** | **作者** | **文献名称** | **出版单位(或归属单位)** | **日期** |
| 1 | 孙鑫 | Vue.js 从入门到实战 | 中国水利水电出版社 | 2020/4 |
| 2 | 马特·弗里斯比 | JavaScript高级程序设计（第4版） | 人民邮电出版社 | 2020/9 |
| 3 |  |  |  |  |
|  |  |  |  |  |

## 术语与缩写解释

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# 假设前提

假设是用户使用该系统：

硬件方面只要一台普通的低配置家用或商用电脑即可，软件方面最好是IE11以上的浏览器

假设是开发人员接手该项目：

环境需求：Node.js、Vue框架、webpack

# 概要设计（总体设计）

*[描述本项目被分配所要实现的所有模块。]*

## 模块功能列表(功能架构)

*[描述本项目的功能模块及分类组织关系。]*

如：

|  |  |  |
| --- | --- | --- |
| **模块名称** | **模块说明** | **备注** |
| *登录* |  |  |
| *注册* |  |  |
| *总结* |  |  |
| *任务* |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 模块间交互图

*[设计人员可以根据实际情况选择使用层次结构图、交互图、状态图、数据流图中一种或者多种来描述本项目所涉及到的各模块之间的交互关系图（至少需要提供一种示意图），并添加必要的文字描述。]*

*如：*

## 模块重用设计【可选】

*[软件重用是利用事先建立好的软件产品创建新软件系统的过程。这个定义蕴含着软件重用所必须包含的两个方面：*

*1. 系统地开发可重用的软件产品。这些软件产品可以是代码，但不应该仅仅局限在代码。必须从更广泛和更高层次来理解，这样才会带来更大的重用收益。比如软件产品还包括：软件分析，设计，测试数据，原型，计划，文档，模板，框架等等。*

*2. 系统地使用这些软件产品作为构筑新的软件产品，来建立新的系统。*

*软件重用的形式（或手段）很多，重用的级别有大有小。主要有以下几种：*

*1. 源代码模块或者类一级的重用。这是最基本的软件重用形式。*

*2. 二进制形式的重用。如组件重用。*

*3. 组装式重用。比如：把好几个应用程序的功能集成在一起。例如，要建立一个门户站点应用，登陆用户既可以查询天气情况，又可以查看股市行情，还可以在线购物。这些功能由不同网络应用服务供应商提供，通过这种组装式重用，就可以非常容易地把上述功能都集成到新的门户站点中。*

*4. 分析级别重用。*

*5. 设计级别重用。*

*6. 软件文档重用。*

*软件重用技术*

*软件重用技术包括: 库函数，模板，面向对象、设计模式、组件、构架。*

*1. 库函数*

*对于库函数的使用者，只要知道函数的名称，返回值的类型, 函数参数和函数功能就可以对其进行调用。*

*2. 面向对象*

*面向对象技术通过方法、消息、类、继承、封装、和实例等机制构造软件系统，并为软件重用提供强有力的支持。如VC++中的MFC。*

*3. 模板*

*模板相当于工业生产中所用的“模具”。有各种各样的模板（如文档模板，网页模板等），利用这些模板可以比较快速地建立对应的软件产品。模板把不变的封装在内部，对可能变化的部分提供了通用接口，由使用者来对这些接口进行设定或实现。*

*4. 设计模式*

*设计模式作为重用设计信息的一种技术，。设计模式描述了在我们周围不断重复发生的问题，该问题的解决方案的核心和解决方案实施的上下文。设计模式命名一种技术并且描述它的成本和收益，共享一系列模式的开发者拥有共同的语言来描述他们的设计。*

*5. 构件*

*构件应从以下几个方面来理解： 1) 构件应是抽象的系统特征单元，具有封装性和信息隐蔽，其功能由它的接口定义。 2) 构件可以是原子的，也可以是复合的。因此它可以是函数，过程或对象类，也可以是更大规模的单元。一个子系统是包含其它构件的构件。 3) 构件是可配置和共享的,这是基于构件开发的基石，且构件之间能相互提供服务。*

*6. 构架*

*构架应从以下几个方面来理解：*

*1) 构架是与设计的同义理解，是系统原型或早期的实现；*

*2) 构架是高层次的系统整体组织；*

*3) 构架是关于特定技术如何合作组成一个特定系统的解释。*

*概要设计主要从库函数，模板，面向对象、设计模式等方面对重用设计进行描述]*

# 数据库设计说明

[说明本项目工作范畴内需要新增、修改及删除的数据表及其相关细节]

## 数据表规划

|  |  |  |
| --- | --- | --- |
| 表名称 | 新增/修改/删除 | 主要存储数据说明 |
| staffUsers | 新增 | 保存用户信息 |
| staffDiary | 新增 | 保存每日总结 |
| task | 新增 | 保存任务信息 |
|  |  |  |

* 1. staffUsers设计说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 说明 |  | | | |
| 字段名称 | 数据类型 | 是否可为空 | 初始化数值 | 备注 |
| userId | INT | 否 | 1 | 用户ID |
| userName | VARCHAR(50) | 否 |  | 用户名 |
| email | VARCHAR(200) | 否 |  | 用户邮箱 |
| psw | VARCHAR(20) | 否 |  | 用户密码 |
| authorityNames | VARCHAR(30) | 否 |  | 权限名 |

* 1. staffDiary设计说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 说明 |  | | | |
| 字段名称 | 数据类型 | 是否可为空 | 初始化数值 | 备注 |
| diaryId | INT | 否 | 1 | 总结ID |
| username | VARCHAR(50) | 否 |  | 用户名 |
| diaryTitle | VARCHAR(100) | 否 |  | 总结标题 |
| diaryContent | VARCHAR(1000) | 否 |  | 总结内容 |
| diaryTime | VARCHAR(60) | 否 |  | 总结时间 |

* 1. task设计说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 说明 |  | | | |
| 字段名称 | 数据类型 | 是否可为空 | 初始化数值 | 备注 |
| taskId | INT | 否 | 1 | 任务ID |
| taskGetName | VARCHAR(100) | 否 |  | 接收人名字 |
| taskSendName | VARCHAR(100) | 否 |  | 发送人名字 |
| taskTitle | VARCHAR(100) | 否 |  | 任务标题 |
| taskContent | VARCHAR(1000) | 否 |  | 任务内容 |
| taskTime | VARCHAR(60) | 否 |  | 任务发布时间 |
| taskFinishTime | VARCHAR(60) | 否 |  | 任务完成时间 |

# 模块1设计说明：登录

## 模块功能描述

该模块是作为登录的，当用户输入完用户名与密码的时候，该模块将信息在数据库中对比，是否存在，如存在就会跳转到主页，不存在就会提示错误。

## 模块类图描述



## 关键方法和过程

当用户输入完用户密码，点击登录按钮后，会执行(submit())函数，该方法会调用login()函数，将信息与数据库中的信息对比。

*[罗列该模块内部的关键或主要方法。]*

|  |  |
| --- | --- |
| 方法名称 | 完成功能 |
| login | 校验 |

### 方法1：login

|  |  |
| --- | --- |
| 方法原型 | login |
| Function |
| 功能说明 | 该方法将输入框的数据与后台数据对比 |
| 调用关系 | 仅作为内部的私有方法 |
| 输入参数 | userName string 用户名 psw string 密码 |
| 输出参数 | code INT 状态码  msg sting 信息  token string token  uusername string 用户名 |
| 返回值 | Object 状态 |
| 备注 | 无 |

#### 函数内部逻辑

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数的实现逻辑,利用流程图并配以流程说明，并建议使用伪码]*

#### 函数设计备注

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数设计的一些特殊考虑，比如基于效率，历史原因，现场运行环境风险而导致实现变更，并给出之后处理建议，保证后续开发的延续性]*

# 模块2设计说明：注册

## 模块功能描述

该模块是作为注册的，当用户输入完用户名、邮箱和密码的时候，该模块将用户名与数据库中信息相对比，是否存在，如存在会提示用户更改用户名，如不存在，会将用户数据保存在数据库中，登录的时候可以使用该数据。

## 模块类图描述



## 关键方法和过程

当用户输入完用户密码，点击登录按钮后，会执行(submit())函数，该方法会调用login()函数，将信息与数据库中的信息对比。

*[罗列该模块内部的关键或主要方法。]*

|  |  |
| --- | --- |
| 方法名称 | 完成功能 |
| login | 校验 |

### 方法1：register

|  |  |
| --- | --- |
| 方法原型 | register |
| Function |
| 功能说明 | 添加用户信息到数据库中 |
| 调用关系 | 仅作为内部的私有方法 |
| 输入参数 | userName string 用户名  email string 邮箱 psw string 密码 |
| 输出参数 | code INT 状态码  msg sting 信息 |
| 返回值 | Object 状态 |
| 备注 | 无 |

#### 函数内部逻辑

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数的实现逻辑,利用流程图并配以流程说明，并建议使用伪码]*

#### 函数设计备注

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数设计的一些特殊考虑，比如基于效率，历史原因，现场运行环境风险而导致实现变更，并给出之后处理建议，保证后续开发的延续性]*

# 模块3设计说明：添加数据

## 模块功能描述

该模块是作为添加数据的，将用户输入的数据添加到数据库中。

## 模块类图描述



## 关键方法和过程

当用户输入完用户密码，点击登录按钮后，会执行(submit())函数，该方法会调用login()函数，将信息与数据库中的信息对比。

*[罗列该模块内部的关键或主要方法。]*

|  |  |
| --- | --- |
| 方法名称 | 完成功能 |
| task | 添加任务 |
| diary | 添加总结 |

### 方法1：task

|  |  |
| --- | --- |
| 方法原型 | task |
| Function |
| 功能说明 | 添加任务到数据库中 |
| 调用关系 | 仅作为内部的私有方法 |
| 输入参数 | taskGetName string 接收人名字  taskTitle string 任务标题  taskTime string 任务时间  taskSendName string 发送人名字  taskFinishTime string 完成时间 |
| 输出参数 | code INT 状态码  msg sting 信息 |
| 返回值 | Object 状态 |
| 备注 | 无 |

#### 函数内部逻辑

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数的实现逻辑,利用流程图并配以流程说明，并建议使用伪码]*

#### 函数设计备注

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数设计的一些特殊考虑，比如基于效率，历史原因，现场运行环境风险而导致实现变更，并给出之后处理建议，保证后续开发的延续性]*

### 方法2：diary

|  |  |
| --- | --- |
| 方法原型 | diary |
| Function |
| 功能说明 | 添加用户总结到数据库中 |
| 调用关系 | 仅作为内部的私有方法 |
| 输入参数 | userName string 用户名  diaryTitle string 总结标题  diaryContent string 总结内容  diaryTime string 完成时间 |
| 输出参数 | code INT 状态码  msg sting 信息 |
| 返回值 | Object 状态 |
| 备注 | 无 |

#### 函数内部逻辑

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数的实现逻辑,利用流程图并配以流程说明，并建议使用伪码]*

#### 函数设计备注

*[此节为可选内容。*

*对于重要的方法/函数，使用本节描述方法/函数设计的一些特殊考虑，比如基于效率，历史原因，现场运行环境风险而导致实现变更，并给出之后处理建议，保证后续开发的延续性]*

# 风险

*[以表格方式指明方案、模块设计的任何已知风险，如果可能，说明减少潜在风险的策略。]*

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 风险 | 严重程度 | 规避措施 |
|  |  |  |  |
|  |  |  |  |

# 附录

*[与本软件设计说明书相关，但不方便在以上章节中包含的内容，或者由于在多个地方需要使用而篇幅较长的信息等，例如，相关的数据辞典，数据结构，方案原型（对比）测试结果等]*