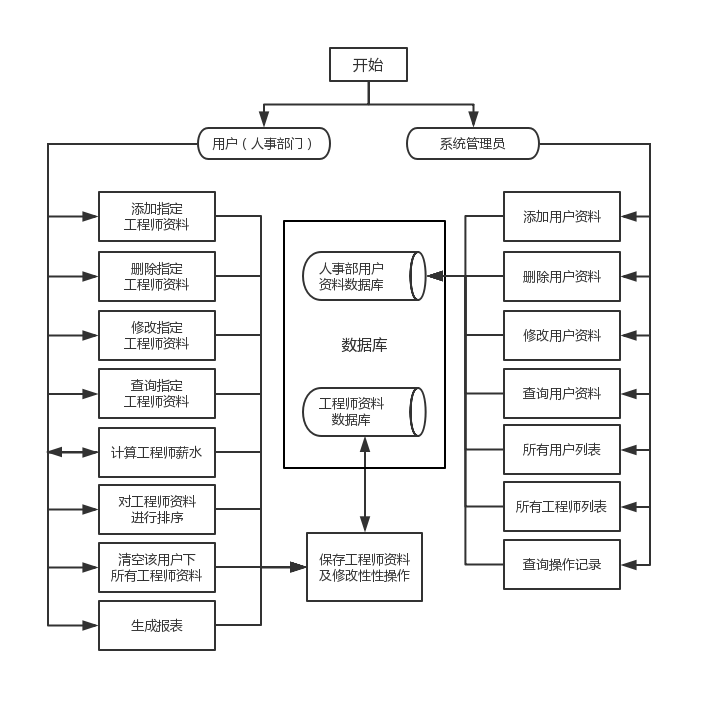
总体设计文档

1. 软件结构图



以下各个接口均是提供给前端工程使用的

1. 模块功能介绍

2.1、登录模块

（1）选择相应的管理权限并输入账号和密码

（2）用户验证进去相应的个人主页，管理员则进入后台管理员模块

提供的接口：111 + 用户名username 输出用户信息

请求数据：111 + 用户名username

返回数据：密码password

2.2、管理员模块

2.2.1、输入用户资料

用户的资料主要包括编号、用户名、密码、姓名、性别。需要对这些输入的信息进行合法性检查。保证系统接收合法的输入。用户输入错误时具有提示功能和重新输入功能。

提供的接口： 222 + insert\_type=1 + 用户信息

请求数据：编号、用户名、密码、姓名、性别

返回数据：是否存储成功

2.2.2、删除指定用户资料

可根据两种方式删除指定用户资料，一是用户的编号，二是用户的用户名。删除资料后，该用户的信息则完全在系统中删除了。其它工程师编号不变。

提供的接口：888 + user\_id/username 删除该user\_id或者username的用户

请求数据：用户的编号(user\_id)或者用户名(user\_name)

返回数据：是否删除成功

调用的接口：删除用户

2.2.3、修改指定用户资料

可根据两种方式找到要修改的用户资料，一是用户的编号，二是用户的用户名。打印原来该用户的信息，同时提示用户进行修改。

提供的接口：444 + insert\_type=1 + 用户user\_id/user\_name + 要修改的数据

请求数据：第1步请求编号(user\_id)或者用户名(user\_name)，第2步请求修改用户资料

返回数据：第一步返回json格式的该用户信息（后台返回），前台处理后，提交后，第二步返回是否修改成功

调用的接口：修改用户

2.2.4、查询指定用户资料

可根据两种方式找到要查询的用户资料，一是用户的编号，二是用户的用户名。打印原来该用户的信息，同时提示用户进行修改。

提供的接口：213 + 用户名/user\_id 输出用户信息

请求数据：编号(user\_id)或者用户名(user\_name)

返回数据：json格式的该用户信息（后台返回），前台处理后，予以显示

调用的接口：查询用户

2.2.5、显示所有操作记录

列出所有的操作详情及时间。

提供的接口：999查询操作记录

请求数据：查询操作记录指令（999）

返回数据：具体用户操作记录、时间、编号

2.2.6、退出系统

当管理员不再使用该系统后，可退出该系统。如果管理员进行了影响用户资料信息的操作，则保存管理员相关操作信息。

2.3、用户模块

2.3.1、输入工程师资料

工程师的资料主要包括编号、姓名、性别、生日、籍贯、学历、地址、电话、工龄、基本薪水。需要对这些输入的信息进行合法性检查。保证系统接收合法的输入。用户输入错误时具有提示功能和重新输入功能。

提供的接口： 222 + insert\_type=2 + 工程师信息

请求数据：编号、姓名、性别、生日、籍贯、学历、地址、电话、工龄、基本薪水

返回数据：是否存储成功

2.3.2、删除指定工程师资料

可根据两种方式删除指定工程师资料，一是工程师的编号，二是工程师的姓名。删除资料后，该工程师的信息则完全在系统中删除了。其它工程师编号不变。由于工程师有同名情况，输入要删除的工程师姓名后，系统列出该用户下所有同名的工程师，用户再选择需要删除的工程师进行删除。

提供的接口：777 + engineer\_id 删除该id的工程师

请求数据：请求工程师的编号(engineer\_id)或者姓名(engineer\_name)以及所需要的接口编号

返回数据：是否删除成功

调用的接口：查询工程师、删除工程师

2.3.3、修改指定工程师资料

可根据两种方式找到要修改的工程师，一是工程师的编号，二是工程师的姓名。打印原来该工程师的信息，同时提示用户进行修改。由于工程师有同名情况，输入要修改的工程师姓名后，系统列出该用户下所有同名的工程师，用户再选择需要修改的工程师进行修改。

提供的接口：444 + insert\_type=2 + 对应engineer\_id + user\_id + 要修改的数据

请求数据：第1步请求555 + 编号(engineer\_id)或者姓名(engineer\_name)，第2步请求444 + insert\_type=2 + 对应engineer\_id + user\_id + 要修改的数据

返回数据：第1步返回json格式的该工程师信息（后台返回），前台处理后，予以显示，第2步返回是否修改成功

调用的接口：修改工程师、查询工程师

2.3.4、查询指定工程师资料

可根据两种方式查询指定工程师资料，一是工程师的编号，二是工程师的姓名。查询后打印该工程师的信息，如果没有该工程师资料则给用户提示。

提供的接口：555 + engineer\_id/engineer\_name + user\_id返回该user\_id用户旗下的对应的engineer\_id的工程师信息

请求数据: 编号(engineer\_id)或者姓名(engineer\_name)

返回数据：json格式的该工程师信息（后台返回）或返回无该工程师的状态

调用的接口：查询工程师

2.3.5、计算工程师薪水

根据当月的月效益，计算工程师的当月工资。在计算机工资时要扣除当月的保险金。程师的月应发的薪水如下计算方法：

薪水＝（基本工资＋10╳月有效工作日天数＋月效益╳工作年限÷100）╳0.9－月保险金。

提供的接口：555 + engineer\_id + user\_id 返回该user\_id用户旗下的对应的engineer\_id的工程师信息

请求数据：工程师的编号、月工作天数、月收益金额、应缴纳保险金

返回数据：计算结果（整型格式）

调用的接口：查询工程师

2.3.6、保存工程师资料

当用户输入工程师资料、删除、修改、排序后需要保存工程师资料。当用户做出这类修改性操作后，要将操作内容、修改内容记录保存至数据库并将具体内容显示在界面上。

请求数据：工程师信息(编号、姓名、性别、生日、籍贯、学历、地址、电话、工龄、基本薪水)

返回数据：是否保存成功的状态

2.3.7、对工程师的资料进行排序

对工程师资料进行排序，排序使用三种方式：编号排序（升序/降序）、姓名排序（升序/降序）和工龄排序（降序/降序）。采用哪种方式，由用户选择。

提供的接口：666 + insert\_type=2 + user\_id 输出所有对应数据

请求数据：user\_id

返回数据：json格式的经排序后的工程师信息（后台返回），前台处理后，予以显示

调用的接口：查询工程师

2.3.8、打印工程师资料信息报表

这时输出工程师除了生日之外的所有资料，包括薪水，以报表的形式输出。提供的接口：666 + user\_id + insert\_type = 2

请求数据：user\_id

返回数据：对应user\_id旗下的json格式的所有工程师信息（后台返回），前台处理后予以显示

调用的接口：查询工程师

2.3.9、清空工程师所有资料

把该用户下所有的工程师资料删除，但不影响其他用户下的工程师信息。其他工程师编号不变。之后可以重新输入工程师资料。

提供的接口：888 + user\_id 给insert\_type赋予空值 删除某用户旗下所有工程师

请求数据：清除指令 + user\_id

返回数据：是否清除成功

2.3.10、退出系统

当用户不再使用该系统后，可退出该系统。并保存用户对工程师资料信息的操作。

1. 数据库设计

|  |  |  |
| --- | --- | --- |
| 1. 成员 | 类型 | 说明 |
| 用户编号 | 整型 | 0<表示长度<=4，自动递增 |
| 用户名 | 字符 | 0<表示长度<=10 |
| 密码 | 字符 | 0<表示长度<=10 |

表一：useraccount用户成员列表

|  |  |  |
| --- | --- | --- |
| 成员 | 类型 | 说明 |
| 用户编号 | 整型 | 0<表示长度<=4，自动递增 |
| 姓名 | 字符 | 0<表示长度<=10 |
| 性别 | 字符 | 0表示女、1表示男 |

表二：userdata用户信息列表

|  |  |  |
| --- | --- | --- |
| 成员 | 类型 | 说明 |
| 编号 | 整型 | 0<表示长度<=4，自动递增 |
| 工龄 | 整型 | 0<=工龄<=50 |
| 工资 | 整型 | 0<=可供输入的长度<=9 |
| 学历 | 整型（枚举） | 1小学 2初中 3中专 4高中 5专科 6本科 7硕士研究生 8博士研究生 |

表三：engineer工程师成员列表

|  |  |  |
| --- | --- | --- |
| 成员 | 类型 | 说明 |
| 编号 | 整型 | 0<表示长度<=4，自动递增 |
| 姓名 | 字符 | 0<表示长度<=10 |
| 性别 | 字符 | 0表示女、1表示男 |
| 出生日期 | 日期 | 格式1999-01-01（1920-2004） |
| 电话 | 整型 | 0<表示长度<=15 |
| 地址 | 字符 | 0<表示长度<=30 |
| 籍贯 | 字符 | 0<表示长度<=10 |

表四：engineerdetails工程师信息列表

|  |  |  |
| --- | --- | --- |
| 成员 | 类型 | 说明 |
| 学历编号 | 整型 | 固定编号 |
| 学历 | 字符 | 1小学 2初中 3中专 4高中 5专科 6本科 7硕士研究生 8博士研究生 |

表五：edu学历列表

|  |  |  |
| --- | --- | --- |
| 成员 | 类型 | 说明 |
| 工程师编号 | 整型 | 0<表示长度<=4，自动递增 |
| 用户编号 | 整型 | 0<表示长度<=4，自动递增 |

表六：mapping映射列表

|  |  |  |
| --- | --- | --- |
| 成员 | 类型 | 说明 |
| 编号 | 整型 | 唯一且自动递增,长度11 |
| 时间 | datetime | 获取所做操作的时间 |
| 数据 | text | 记录某系统用户所做的具体操作，包括管理员 |

表七：records操作记录列表

#创建用户数据库

CREATE DATABASE webservice

USE webservice;

#Table structure for table `edu`

DROP TABLE IF EXISTS `edu`;

CREATE TABLE `edu` (

`education\_id` int(11) NOT NULL AUTO\_INCREMENT,

`education` varchar(10) DEFAULT NULL,

PRIMARY KEY (`education\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=10 DEFAULT CHARSET=utf8;

-- Dumping data for table `edu`

LOCK TABLES `edu` WRITE;

INSERT INTO `edu` VALUES (1,'小学'),(2,'初中'),(3,'中专'),(4,'高中'),(5,'专科'),(6,'本科'),(7,'硕士研究生'),(9,'博士研究生');

UNLOCK TABLES;

#Table structure for table `engineer`

DROP TABLE IF EXISTS `engineer`;

CREATE TABLE `engineer` (

`engineer\_id` int(11) NOT NULL AUTO\_INCREMENT,

`workingYears` int(11) DEFAULT NULL,

`salary` int(11) NOT NULL,

`education\_id` int(11) DEFAULT NULL,

PRIMARY KEY (`engineer\_id`),

KEY `engineerdata\_edu\_education\_id\_fk` (`education\_id`),

CONSTRAINT `engineerdata\_edu\_education\_id\_fk` FOREIGN KEY (`education\_id`) REFERENCES `edu` (`education\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=utf8;

-- Dumping data for table `engineer`

LOCK TABLES `engineer` WRITE;

INSERT INTO `engineer` VALUES (1,5,8000,7),(3,5,9000,5),(4,1,11000,6);

UNLOCK TABLES;

#Table structure for table `engineerdetails`

DROP TABLE IF EXISTS `engineerdetails`;

CREATE TABLE `engineerdetails` (

`engineer\_id` int(11) NOT NULL,

`name` varchar(10) DEFAULT NULL,

`sex` tinyint(1) DEFAULT NULL,

`birthday` date DEFAULT NULL,

`mobile` int(11) DEFAULT NULL,

`address` varchar(50) DEFAULT NULL,

`nativeplace` varchar(50) DEFAULT NULL,

PRIMARY KEY (`engineer\_id`),

CONSTRAINT `engineerDetails\_engineerdata\_engineer\_id\_fk` FOREIGN KEY (`engineer\_id`) REFERENCES `engineer` (`engineer\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Dumping data for table `engineerdetails`

LOCK TABLES `engineerdetails` WRITE;

INSERT INTO `engineerdetails` VALUES (1,'李四',1,'2020-03-31',189765131,'浙江省杭州市','浙江省温州市'),(3,'陈二',2,'1998-12-11',156478953,'浙江温州','浙江杭州'),(4,'陈三',1,'1989-11-11',165315756,'浙江温州','浙江宁波');

UNLOCK TABLES;

#Table structure for table `mapping`

DROP TABLE IF EXISTS `mapping`;

CREATE TABLE `mapping` (

`engineer\_id` int(11) NOT NULL,

`user\_id` int(11) NOT NULL,

PRIMARY KEY (`engineer\_id`,`user\_id`),

KEY `mapping\_account\_user\_id\_fk` (`user\_id`),

CONSTRAINT `mapping\_account\_user\_id\_fk` FOREIGN KEY (`user\_id`) REFERENCES `useraccount` (`user\_id`),

CONSTRAINT `mapping\_engineerdata\_engineer\_id\_fk` FOREIGN KEY (`engineer\_id`) REFERENCES `engineer` (`engineer\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Dumping data for table `mapping`

LOCK TABLES `mapping` WRITE;

INSERT INTO `mapping` VALUES (1,1),(3,1),(4,1);

UNLOCK TABLES;

#Table structure for table `records`

DROP TABLE IF EXISTS `records`;

CREATE TABLE `records` (

`data` text,

`time` datetime DEFAULT NULL,

`id` int(11) NOT NULL AUTO\_INCREMENT,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Dumping data for table `records`

LOCK TABLES `records` WRITE;

UNLOCK TABLES;

#Table structure for table `useraccount`

DROP TABLE IF EXISTS `useraccount`;

CREATE TABLE `useraccount` (

`user\_id` int(11) NOT NULL AUTO\_INCREMENT,

`userName` varchar(10) NOT NULL,

`password` varchar(16) NOT NULL,

PRIMARY KEY (`user\_id`),

UNIQUE KEY `account\_userName\_uindex` (`userName`)

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=utf8;

-- Dumping data for table `useraccount`

LOCK TABLES `useraccount` WRITE;

INSERT INTO `useraccount` VALUES (1,'929259187','991231'),(2,'123456','88888888');

UNLOCK TABLES;

#Table structure for table `userdata`

DROP TABLE IF EXISTS `userdata`;

CREATE TABLE `userdata` (

`user\_id` int(11) NOT NULL,

`name` varchar(10) DEFAULT NULL,

`sex` tinyint(1) DEFAULT NULL,

PRIMARY KEY (`user\_id`),

CONSTRAINT `userdata\_account\_user\_id\_fk` FOREIGN KEY (`user\_id`) REFERENCES `useraccount` (`user\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Dumping data for table `userdata`

LOCK TABLES `userdata` WRITE;

INSERT INTO `userdata` VALUES (1,'张三',1),(2,'陈一',1);

UNLOCK TABLES;

1. 实现计划

时间进度安排

第1周：分组

第2周：在github.com上建立仓库

第3-5周：设计并完善需求分析文档

第6-7周：设计并完善总体设计文档，设计数据库及软件后台

第7-13周：完成软件功能及页面，并完善所有内容

第14周：软件测试