|  |  |
| --- | --- |
|  | |
| **武汉大学计算机学院**  **大型应用软件课程设计**  图片  **《银行监控告警系统》**  **项目概要设计**  **V1.0** | |
|  |  |

**目录**

[第一部分 引言 1](#_Toc27723)

[一、编写目的 1](#_Toc6824)

[二、读者对象 1](#_Toc10484)

[三、术语与缩写解释 1](#_Toc14868)

[四、参考资料 1](#_Toc31258)

[第二部分 项目概述 1](#_Toc10630)

[一、项目描述 1](#_Toc12222)

[二、项目功能描述 2](#_Toc30417)

[1、用户登录 2](#_Toc3795)

[2、账号管理（管理员） 2](#_Toc20424)

[3、查看实时数据 2](#_Toc16363)

[（1）查看ATM与外汇交易金额实时趋势变化 2](#_Toc11442)

[（2）查看ATM与外汇交易地点实时分布情况 3](#_Toc27706)

[（3）查看外汇买入目的实时趋势变化 3](#_Toc9145)

[（4）查看外汇买入币种占比实时变化 3](#_Toc17385)

[（5）查看外汇买入年龄段占比实时变化 4](#_Toc21627)

[（5）查看实时交易笔数和金额 4](#_Toc16337)

[4、查看离线数据 4](#_Toc5468)

[5、监控数据可视化 6](#_Toc16901)

[6、监控告警信息 7](#_Toc6232)

[7、设置参数 8](#_Toc5094)

[第三部分 设计约束 8](#_Toc11100)

[一、需求约束 8](#_Toc30093)

[1、本系统应当遵循的技术标准 8](#_Toc6257)

[1.1 命名规则 8](#_Toc20943)

[1.2 注释规则 10](#_Toc3138)

[1.3 前后端交互规则 10](#_Toc817)

[2、软、硬件环境标准 12](#_Toc23998)

[3、接口/协议标准 12](#_Toc14709)

[4、用户界面标准 13](#_Toc10218)

[5、软件质量 13](#_Toc19631)

[二、隐含约束 13](#_Toc28445)

[第四部分 银行监控告警方案设计 14](#_Toc29199)

[一、安全设计 14](#_Toc5432)

[二、银行监控告警系统业务流程 14](#_Toc10160)

[1、管理员管理员工账户流程 14](#_Toc15522)

[2、实时数据分析流程 15](#_Toc6192)

[3、离线数据分析流程 16](#_Toc30727)

[4、监控告警业务流程 17](#_Toc31724)

[三、业务功能概要结构 18](#_Toc21806)

[1、账户管理模块 19](#_Toc27908)

[银行运维系统管理员角色： 19](#_Toc866)

[1.1 新建本银行(本部门)账户 19](#_Toc3221)

[1.2 删除账户 19](#_Toc6723)

[2、实时数据分析模块 19](#_Toc2774)

[2.1、ATM与外汇金额实时趋势变化分析 20](#_Toc27639)

[2.2、ATM与外汇交易地点分布情况分析 20](#_Toc28915)

[2.3、实时外汇买入目的分布情况分析 21](#_Toc26134)

[2.4、实时外汇买入币种分布情况分析 21](#_Toc17110)

[2.3、实时外汇买入年龄段分布情况分析 21](#_Toc24264)

[3、离线数据分析模块 22](#_Toc76)

[3.1、过去一天 ATM 取款金额分布情况分析 22](#_Toc22502)

[3.2、过去一天ATM取款地点分布情况分析 22](#_Toc2074)

[3.3、过去一天外汇买入目的占比分析 23](#_Toc20516)

[3.4、过去一天外汇买入币种占比分析 23](#_Toc13574)

[3.5、过去一天外汇买入年龄段占比分析 23](#_Toc14989)

[3.6、过去一天外汇买入地点分布情况分析 24](#_Toc3042)

[4、监控告警模块 24](#_Toc14038)

[4.1、参数设置 24](#_Toc19215)

[4.2、展示ATM取款实时交易量 25](#_Toc14677)

[4.3、展示ATM取款实时响应时间 25](#_Toc28448)

[4.4、展示外汇买入实时交易量 25](#_Toc1948)

[4.5、展示外汇买入实时响应时间 26](#_Toc7549)

[4.6、展示ATM取款实时交易趋势图 26](#_Toc22284)

[4.7、展示外汇买入实时交易趋势图 26](#_Toc11576)

[4.8、展示ATM取款实时响应时间趋势图 26](#_Toc1243)

[4.9、展示外汇买入实时响应时间趋势图 27](#_Toc1906)

[4.10、告警信息处理 27](#_Toc23387)

[四、模块定义 27](#_Toc9220)

[第五部分 E-R实体设计 29](#_Toc13968)

[一、E-R实体结构图 29](#_Toc19523)

[二、实体描述 30](#_Toc10835)

[1、用户（User）实体描述 30](#_Toc23835)

[2、实时交易总金额（RealTimeTransactionAmount）实体描述 30](#_Toc7818)

[3、实时交易地区分布 31](#_Toc19161)

[（RealTimeTransactionProvince）实体描述 31](#_Toc14921)

[4、实时外汇币种（RealTimeFXCurrency）实体描述 31](#_Toc31827)

[5、实时外汇目的（RealTimeFXPurpose）实体描述 32](#_Toc31959)

[6、实时外汇年龄（RealTimeFXAge）实体描述 32](#_Toc87)

[7、离线交易金额分布（OfflineTransactionAmount）实体描述 32](#_Toc7140)

[8、离线交易地区（OfflineTransactionArea）实体描述 33](#_Toc30596)

[9、离线外汇币种占比（OfflineFXCurrency）实体描述 33](#_Toc14460)

[10、离线外汇目的占比（OfflineFXPurpose）实体描述 33](#_Toc7676)

[11、离线外汇年龄占比（OfflineFXAge）实体描述 34](#_Toc18548)

[12、响应时间监控（NodeDurationMonitor）实体描述 34](#_Toc31631)

[第六部分 总体设计 35](#_Toc16489)

[一、银行监控告警平台逻辑架构设计 35](#_Toc13900)

[1、数据采集层 36](#_Toc10885)

[1.1、Flume 36](#_Toc8863)

[1.2、Hadoop 37](#_Toc30891)

[1.3、Kafka 38](#_Toc17770)

[2、数据存储层 38](#_Toc5301)

[2.1、Hive集群 39](#_Toc24495)

[2.2、Spark集群 39](#_Toc2020)

[3、数据计算与分析层 39](#_Toc6345)

[3.1、离线分析 39](#_Toc6850)

[3.2、实时分析 40](#_Toc17229)

[4、平台应用层 40](#_Toc6392)

[5、展示层 40](#_Toc23043)

[二、物理架构设计 40](#_Toc231)

[三、技术架构设计 41](#_Toc32157)

[1、MVC模型设计 41](#_Toc25866)

[2、大数据模型设计 43](#_Toc16649)

[2.1、Azkaban 43](#_Toc13673)

[2.2、Flume 43](#_Toc16800)

[2.3、Hadoop 44](#_Toc6567)

[2.4、Hive 45](#_Toc12669)

[2.5、Kafka 46](#_Toc2157)

[2.6、Spark Streaming 47](#_Toc28086)

[3、SpringBoot 47](#_Toc28480)

[4、Vue 48](#_Toc18268)

[5、Mybatis 48](#_Toc5644)

[第七部分 用户界面设计 49](#_Toc7522)

[一、页面布局设计 49](#_Toc990)

[1、WEB 登录界面： 49](#_Toc5856)

[2、创建用户界面： 50](#_Toc5801)

[3、用户信息界面： 50](#_Toc2126)

[4、实时业务指标分析界面： 51](#_Toc19985)

[5、离线业务指标分析界面： 52](#_Toc9125)

[5、监控指标页面： 52](#_Toc2434)

[6、设置参数页面： 53](#_Toc21135)

[7、警告信息页面： 54](#_Toc15401)

[第八部分 运行环境和部署 55](#_Toc21258)

[一、运行环境 55](#_Toc28038)

[1、故障监控服务器环境 55](#_Toc12841)

[2、客户机器环境 55](#_Toc21417)

[3、开发环境要求 56](#_Toc24026)

[二、系统性能要求 56](#_Toc31847)

**第一部分 引言**

**一、编写目的**

编写本文的主要目的是把需求分析得到的用例模型转换为软件结构和数据结构。设计软件结构的具体任务是：将一个复杂系统按功能进行模块划分、建立模块的层次结构及调用关系、确定模块间的接口及人机界面等。数据结构设计包括数据特征的描述、确定数据的结构特性、以及数据库的设计。

本设计是指导详细设计和项目实施的重要指导性文件，也是进行系统集成测试和重要依据。

**二、读者对象**

该文档的读者为用户代表、软件分析人员、开发管理人员和测试人员。

**三、术语与缩写解释**

1、BWS: Bank-Warning System，即银行监控告警系统

**四、参考资料**

《银行监控告警系统-用户需求说明》

**第二部分 项目概述**

**一、项目描述**

随着大数据技术的成熟，银行业开始探索以大数据的方式解决运维领域中进行快速故障定位的问题，以提高银行运维的技术水平，减少故障的解决时间。

本项目对银行ATM取款业务和外汇买入业务进行全链路监控的解决方案 通过该方案的实施，可以对应用进行精准的数据收集和监控，结合可视化和数据分析，能够实现对应用的故障进行快速发现和定位，在实际生产工作中极大地提高了效率。

**二、项目功能描述**

**1、用户登录**

用户登录是当用户完成注册后按照对应账号密码进入网页应用的位置。

用户必须填写以下信息以完成登录：

（1)   用户账号

（2)   密码

使用文本框输出信息，通过登录按钮进行登录，登录之后跳转到离线数据分析界面。

**2、账号管理（管理员）**

银行运维管理员进入账号管理界面，能够查看到本银行所有运维人员的账号信息。

管理员能在此界面：

（1）删除账号

（2）新建账号

**3、查看实时数据**

**（1）查看ATM与外汇交易金额实时趋势变化**

用户进入实时数据分析界面，通过折线图直观查看ATM与外汇金额交易趋势变化

折线图包括如下信息：

1）以当天的每个整点时刻、或者当前每分钟为横轴自变量（可选择）

2）以交易金额为纵轴

折线图数据每5秒更新一次

**（2）查看ATM与外汇交易地点实时分布情况**

用户进入实时数据分析界面，通过地图直观查看ATM与外汇交易地点分布情况。

地图包括如下信息：

1）不同色系代表不同交易种类，黄色代表ATM，橙色代表外汇，以圆圈色块显示

2）以圆圈色块大小代表交易金额总量

地图数据每5秒更新一次

**（3）查看外汇买入目的实时趋势变化**

用户进入实时数据分析界面，通过点击不同环形图查看不同外汇买入目的的实时占比情况

1)买入目的种类如下：

a)旅游 b)就医 c)留学 d)教育 e)其他

2)不同颜色显示不同的种类，富文本框显示具体金额和百分比情况

显示最近五秒的实时数据。

**（4）查看外汇买入币种占比实时变化**

用户进入实时数据分析界面，通过点击圆饼图查看外汇买入币种的占比对比实时变化情况

1)外汇买入币种有以下几类：

a)美元 b)欧元 c)日元 d)加拿大元 e)英镑 f)港币

2）不同颜色显示不同的种类，富文本框显示具体金额和百分比情况

显示最近五秒的实时数据。

**（5）查看外汇买入年龄段占比实时变化**

用户进入实时数据分析界面，通过点击圆饼图查看实时外汇买入客户年龄占比情况

1）年龄分为如下几类：

a)18-25 b)26-35 c)36-45 d)46-60 e)61-80 f)81以上

2）不同颜色显示不同的种类，富文本框显示具体金额和百分比情况

显示最近五秒的实时数据

**（5）查看实时交易笔数和金额**

用户进入实时数据分析界面，通过右上框看到实时两种交易金额和笔数。

**4、查看离线数据**

**（1）查看过去一天ATM取款金额分布情况**

用户进入离线数据分析页面查看过去一天银行统计的 ATM 取款金额分布情况，分为如下几类：

（1）小于等于100 元

（2）101-500 元

（3）501-1000 元

（4）1001-10000 元

（5）10001 元或以上

通过条形图直观显示 ATM 取款金额分布。

**（2）查看过去一天ATM取款地点分布情况**

用户进入离线数据分析页面查看过去一天中国境内的银行统计的 ATM 取款地点分布情况，结合当地 ATM 取款金额分类的占比，由圆圈显示在地图上：交易金额总量越大圆圈越大，颜色由黄色显示。

**（3）查看过去一天外汇买入目的占比**

用户进入离线数据分析页面查看过去一天外汇买入目的占比情况，分为如下几类：

（1）旅游

（2）就医

（3）留学

（4）教育

（5）其他

通过环形图直观展示占比情况，用不同颜色显示。点击不同颜色的板块可以在一旁显示对应的币种和买入总金额和所占百分比信息。

**（4）查看过去一天外汇买入币种占比**

用户进入离线数据分析页面查看过去一天外汇买入目的占比情况，分为如下几类：

（1）美元

（2）欧元

（3）日元

（4）加拿大元

（5）英镑

（6）港币

通过饼状图直观展示占比情况，用不同颜色表示。点击不同颜色的板块可以在一旁显示对应的币种和买入总金额和所占百分比信息。

**（5）查看过去一天外汇买入年龄段占比**

用户进入离线数据分析页面查看过去一天外汇买入目的占比情况，分为如下几类：

（1）18-25

（2）26-35

（3）36-45

（4）46-60

（5）61-80

（6）81 或以上

通过饼状图直观展示占比情况，用不同颜色显示。点击不同颜色的板块可以在一旁显示对应的年龄段，买入总金额和所占百分比信息。

**（6）查看过去一天外汇买入地点情况**

用户进入离线数据分析页面由地图显示，与 ATM 取款金额显示在一张图上，橙色显示，查看过去一天中国境内的银行统计的外汇买入地点分布情况，结合当地外汇买入的总金额情况，由圆圈显示在地图上：交易金额总量越大圆圈越大。

**5、监控**数据可视化

监控数据可视化是按照实时(每一小时)记录两类交易包括ATM取款、外汇买入的数据情况。

包括如下两种情况的可视化数据：

1.    实时指标

    （1） ATM取款实时响应时间

    （2） 外汇买入实时响应时间

2.    趋势图

    （1） ATM取款各节点最近五分钟响应时间

    （2） 外汇买入各节点最近五分钟响应时间

根据请示参数中的Status参数值，将实时指标中相应KPI的字体颜色改变。

同时在趋势图上，将相应的时间点颜色改变。

空心点代表Normal

红色实心点代表Warning

6、监控告警信息

监控告警信息是根据两类交易包括ATM取款、外汇买入的响应时间因超出预期值而发出的警告信息。点击处理按钮之后该条目在刷新后不会出现在列表中。

监控告警信息列表包括以下条目：

（1）  告警时间

（2） 告警节点：  ATMEntry、ATMServer、 CoreService、  FXEntry、  FXServer、  FXCoreEntry

（3） 响应时间

（4） 处理故障

还包括一个展示最近一小时各节点响应警告率的条形图：

（1）横坐标为六个节点

（2）纵坐标为警告率百分比

**7、设置参数**

银行管理员进入参数设置页面，可以对本银行的模拟交易参数进行设置，包括：

ATMEntry、ATMServer、ATMService、FXEntry、FXServer、CoreEnry、CoreService 的响应时间，ATM 取款每分钟交易量和外汇买入每分钟交易量。

**第三部分 设计约束**

**一、需求约束**

**1、本系统应当遵循的技术标准**

1.1 命名规则

1、类名使用 UpperCamelCase 风格，但以下情形例外：DO / BO / DTO / VO / AO /

PO / UID 等。

正例：ForceCode / UserDO / HtmlDTO / XmlService / TcpUdpDeal / TaPromotion

反例：forcecode / UserDo / HTMLDto / XMLService / TCPUDPDeal / TAPromotion

2、方法名、参数名、成员变量、局部变量都统一使用 lowerCamelCase 风格。

正例： localValue / getHttpMessage() / inputUserId

3、常量命名全部大写，单词间用下划线隔开，力求语义表达完整清楚，不要嫌名字长。

正例：MAX\_STOCK\_COUNT / CACHE\_EXPIRED\_TIME

反例：MAX\_COUNT / EXPIRED\_TIME

4、抽象类命名使用 Abstract 或 Base 开头；异常类命名使用 Exception 结尾；测试类

命名以它要测试的类的名称开始，以 Test 结尾。

5、类型与中括号紧挨相连来表示数组。

正例：定义整形数组 int[] arrayDemo。

反例：在 main 参数中，使用 String args[]来定义。

6、POJO 类中的任何布尔类型的变量，都不要加 is 前缀，否则部分框架解析会引起序列 化错误。

说明：在本文 MySQL 规约中的建表约定第一条，表达是与否的变量采用 is\_xxx 的命名方式，所以，需要在<resultMap>设置从 is\_xxx 到 xxx 的映射关系。

反例：定义为基本数据类型 Boolean isDeleted 的属性，它的方法也是 isDeleted()，框架在反向解析的时候，“误以为”对应的属性名称是 deleted，导致属性获取不到，进而抛出异常。

7、包名统一使用小写，点分隔符之间有且仅有一个自然语义的英语单词。包名统一使用单数形式，但是类名如果有复数含义，类名可以使用复数形式。

正例：应用工具类包名为 com.bws.ei.kunlun.aap.util、类名为 MessageUtils（此规则参考 spring 的框架结构）

1.2 注释规则

1、所有的类都必须添加创建者和创建日期。

说明：在设置模板时，注意 IDEA 的@author 为`${USER}`，日期的设置统一为 yyyy/MM/dd 的格式。

正例：

/\*\*

\* @author yangguanbao

\* @date 2016/10/31

\*/

2、方法内部单行注释，在被注释语句上方另起一行，使用//注释。方法内部多行注释使

用/\* \*/注释，注意与代码对齐。

1.3 前后端交互规则

1、【强制】前后端交互的 API，需要明确协议、域名、路径、请求方法、请求内容、状态码、响

应体。

说明：

1） 协议：生产环境必须使用 HTTPS。

2） 路径：每一个 API 需对应一个路径，表示 API 具体的请求地址：

a） 代表一种资源，只能为名词，推荐使用复数，不能为动词，请求方法已经表达动作意义。

b） URL 路径不能使用大写，单词如果需要分隔，统一使用下划线。

c） 路径禁止携带表示请求内容类型的后缀，比如".json",".xml"，通过 accept 头表达即可。

3） 请求方法：对具体操作的定义，常见的请求方法如下：

a） GET：从服务器取出资源。

b） POST：在服务器新建一个资源。

c） PUT：在服务器更新资源。

d） DELETE：从服务器删除资源。

4） 请求内容：URL 带的参数必须无敏感信息或符合安全要求；body 里带参数时必须设置 Content-Type。

5） 响应体：响应体 body 可放置多种数据类型，由 Content-Type 头来确定

2、【强制】服务端发生错误时，返回给前端的响应信息必须包含 HTTP 状态码，errorCode、errorMessage、用户提示信息四个部分。

说明：四个部分的涉众对象分别是浏览器、前端开发、错误排查人员、用户。其中输出给用户的提示信息

正例：常见的 HTTP 状态码如下

1） 200 OK: 表明该请求被成功地完成，所请求的资源发送到客户端。

2） 401 Unauthorized: 请求要求身份验证，常见对于需要登录而用户未登录的情况。

3） 403 Forbidden：服务器拒绝请求，常见于机密信息或复制其它登录用户链接访问服务器的情况。

4） 404 Not Found: 服务器无法取得所请求的网页，请求资源不存在。

5） 500 Internal Server Error: 服务器内部错误

3、【强制】在前后端交互的 JSON 格式数据中，所有的 key 必须为小写字母开始的

lowerCamelCase 风格，符合英文表达习惯，且表意完整。

正例：errorCode / errorMessage / assetStatus / menuList / orderList / configFlag

反例：ERRORCODE / ERROR\_CODE / error\_message / error-message / errormessage /

ErrorMessage / msg

4、【推荐】服务端返回的数据，使用 JSON 格式而非 XML。

**2、软、硬件环境标准**

用VSCode编写前端，IDEA编写后端，数据库使用MySQL，在Linux环境下运行模拟交易程序。

**3、接口/协议标准**

采用HTTP协议进行数据交互

**4、用户界面标准**

采用ECharts标准库和Element UI设计

**5、软件质量**

1）正确性

系统必须交易能够被正确处理；

2）健壮性

系统应能够7\*24小时无故障运行；

3）效率性

系统可以支持页面每分钟刷新，数据库数据每分钟更新；

4）易用性

界面应采用图形化操作方式，便于业务人员操作；

5）可扩展性

应该充分考虑到将来交易的修改或增加，避免需求变更时大规模修改程序。

**二、隐含约束**

1）用户具有基本的业务技能和基本的电脑知识，对我们提供的操作界面应保证他们经过简单培训后无障碍的操作；

2）软件可以流畅运行在Win10版本的设备中；

3）应该把有可能变动的参数存放到配置文件或数据库中，保证修改参数的灵活性。

**第四部分 银行监控告警方案设计**

**一、安全设计**

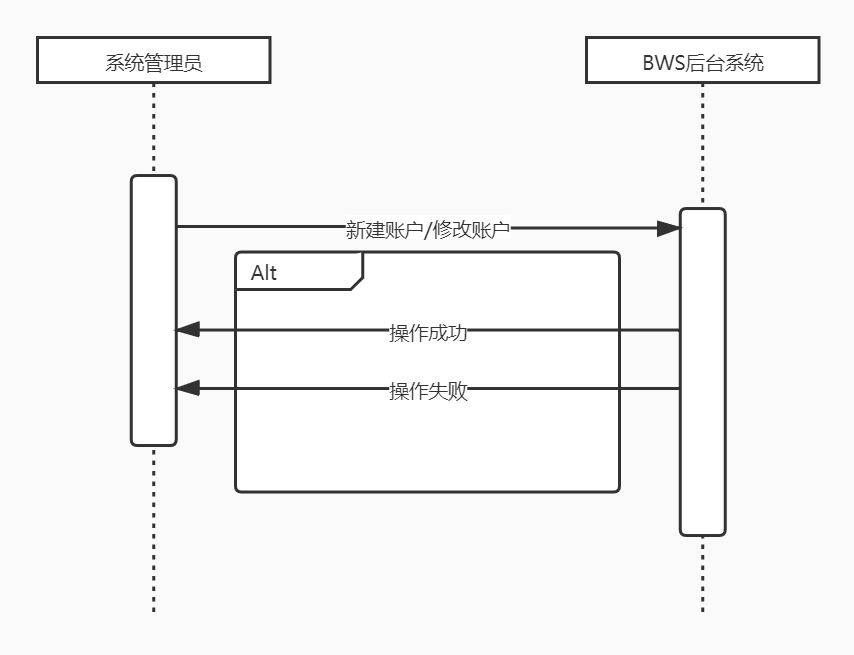
（1）管理员添加用户之后，分配初始密码默认为123456，但是之后在管理界面看不到用户的密码。

（2）用户登录需要填写随机生成的验证码。

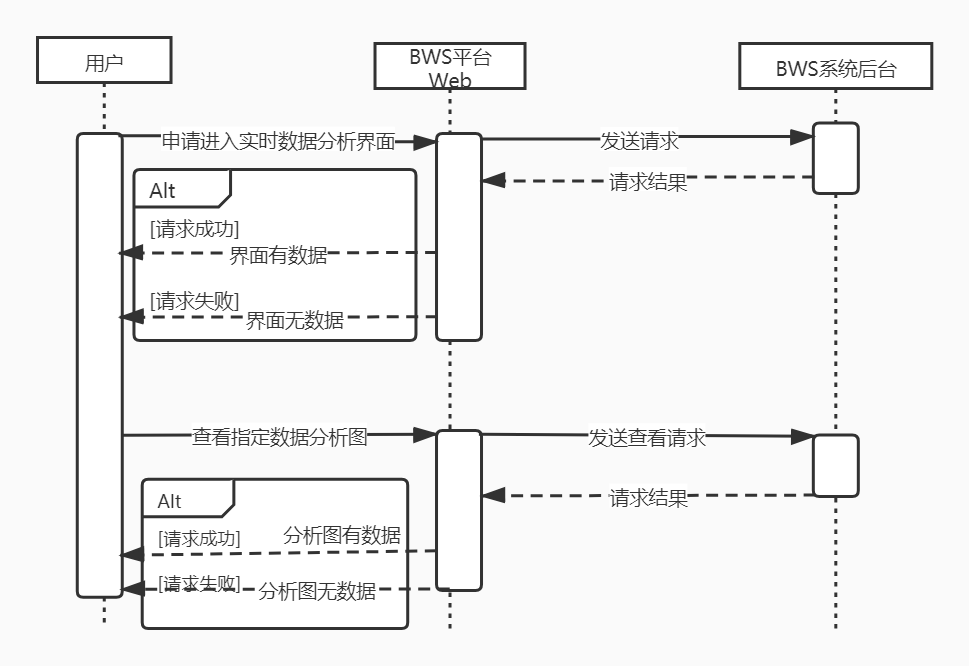
（3）添加防火墙防止数据库被非法侵入。

**二、银行监控告警系统业务流程**

1、管理员管理员工账户流程



**2、实时数据分析流程**



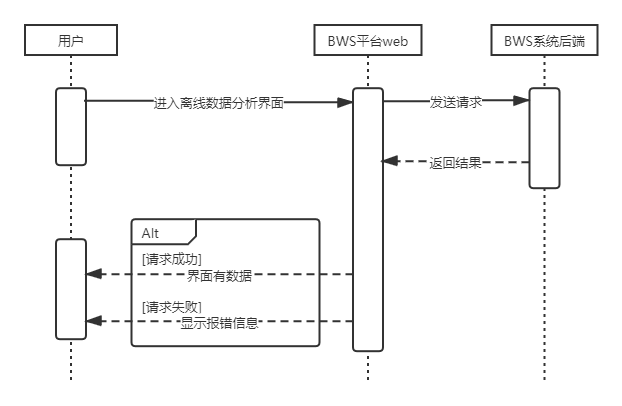
**说明：**实时数据分析业务流程包括：

1)用户申请进入实时数据分析界面

2)平台向后台发送进入请求，后台调用数据库信息并返回请求结果

3)平台跳转到实时数据分析界面，成功则界面上有实时数据分析图。失败则无数据并在界面提示错误信息

**3、离线数据分析流程**



**说明：**离线数据分析业务流程包括：

1)用户申请进入离线数据分析界面

2)平台向后台发送进入请求，后台调用数据库信息并返回请求结果

3)平台跳转到离线数据分析界面，成功则界面上有离线数据分析图。失败则界面上的分析图无数据，并显示出错信息。

4)用户根据想要查看的数据点击界面的相应分析图板块进行查看（如外汇买入目的环形图，外汇买入币种圆饼图，外汇买入客户年龄图圆饼图等）

**4、监控告警业务流程**

|  |
| --- |
| 图片 |

**说明：**监控告警分析业务流程包括：

1)用户申请进入监控指标界面

2)平台向后台发送进入请求，后台调用数据库信息并返回请求结果

3)平台跳转到监控指标界面，成功则界面上有实时指标及趋势图。失败则无数据并在界面提示错误信息

4)平台跳转到监控故障信息列表，成功则界面上有监控故障信息。失败则无数据并在界面提示错误信息

**三、业务功能概要结构**

**1、账户管理模块**

**银行运维系统管理员角色：**

**1.1 新建本银行(本部门)账户**

1.管理员登陆系统

2.进入账户管理页面

3.点击新增账户

4.输入新增账户的用户名,密码和其他个人信息

5.点击完成，成功新建一个系统账户

1.2 删除账户

1.系统管理员登陆系统

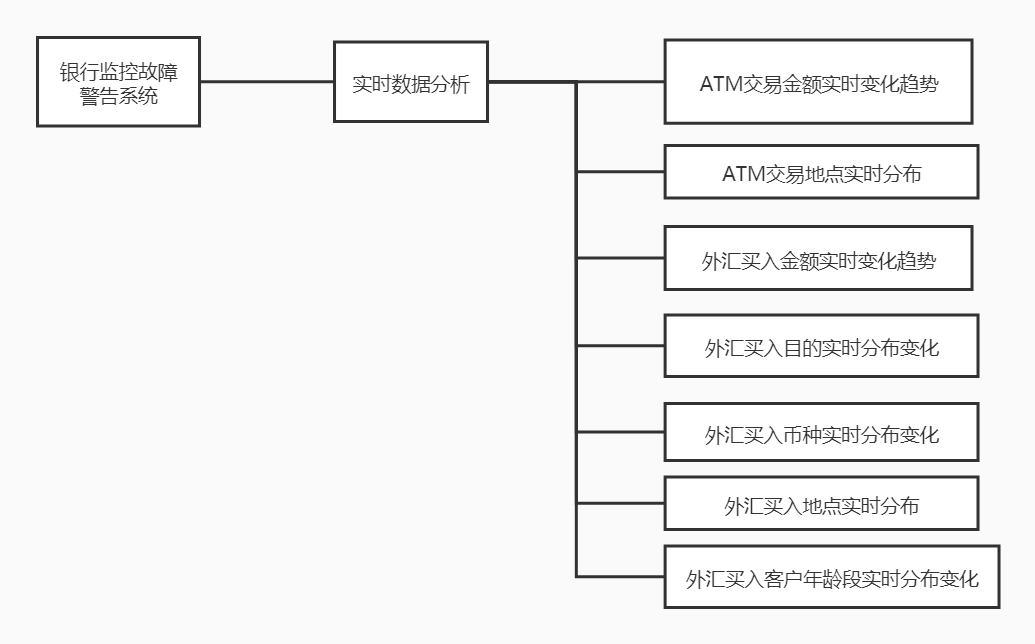
2.进入账户管理页面

3.在展示出的账户列表选择将要删除的账户

4.点击(批量)删除

5.完成删除操作

**2、实时数据分析模块**



**2.1、ATM与外汇金额实时趋势变化分析**

1.初始化实时数据分析界面

2.系统向后台请求基本数据：截止当前的ATM和外汇流水数据，每小时或每分钟的ATM交易金额总额，每小时的外汇买入金额总额

3.系统显示数据分析结果，使用折线图显示

**2.2、ATM与外汇交易地点分布情况分析**

1.初始化实时数据分析界面

2.系统向后台请求基本数据：截止当前的ATM和外汇流水数据，截止当前的一个小时内或每分钟的每个地方的ATM交易金额总额，截止当前的一个小时内或每分钟的每个地方的外汇交易金额总额

3.系统显示数据分析结果，使用地图显示

**2.3、实时外汇买入目的分布情况分析**

1.初始化实时数据分析界面

2.用户点击对应外汇买入目的环形图

3.系统向后台请求基本数据：截止当前的一个小时内或每分钟的外汇买入目的金额占比百分数（已排序）

4.系统显示数据分析结果，使用环形图显示

**2.4、实时外汇买入币种分布情况分析**

1.初始化实时数据分析界面

2.用户点击外汇买入币种占比饼状图

3.系统向后台请求基本数据：截止当前的一个小时内或每分钟的外汇买入币种占比数据（已排序），各币种的交易金额数，各币种所占百分比，占比最大的币种

4.系统显示数据分析结果，使用饼状显示

**2.3、实时外汇买入年龄段分布情况分析**

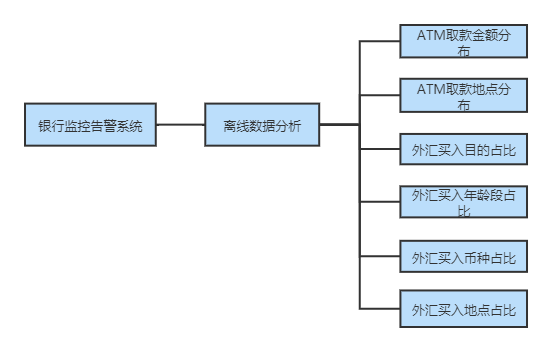
1.初始化实时数据分析界面

2.用户点击外汇买入客户年龄段占比饼状图

3.系统向后台请求基本数据：截止当前的一个小时内或每分钟的外汇买入客户年龄段占比数据（已排序），各年龄段交易的总金额，各年龄段占比百分数，占比最大的年龄段

4.系统显示数据分析结果，使用饼状显示

**3、离线数据分析模块**



**3.1、过去一天 ATM 取款金额分布情况分析**

1、初始化离线数据分析可视化界面

2、系统向服务器发起分析数据请求。

3、显示过去一天的ATM取款金额分布结果，用条形图显示，条柱上显示百分比。

**3.2、过去一天ATM取款地点分布情况分析**

1、初始化离线数据分析可视化界面

2、系统向服务器发起分析数据请求。

3、显示过去一天的ATM取款地点分布结果，在地图上用蓝色圆圈显示。

**3.3、过去一天外汇买入目的占比分析**

1、初始化离线数据分析可视化界面

2、系统向服务器发起分析数据请求。

3、显示过去一天的外汇买入目的占比结果，用环形图显示，占比最多的一项用红色显示，最少用绿色显示，用光标点击不同颜色的环形图可以查看具体情况。

**3.4、过去一天外汇买入币种占比分析**

1、初始化离线数据分析可视化界面

2、系统向服务器发起分析数据请求。

3、显示过去一天的外汇买入币种占比情况。通过饼状图直观展示占比情况，占比最多的一项用红色显示，最少的一项用绿色显示，其他用随机颜色表示。点击不同颜色的板块可以在一旁显示对应的币种和买入总金额和所占百分比信息，默认显示最大占比币种信息。

**3.5、过去一天外汇买入年龄段占比分析**

1、初始化离线数据分析可视化界面

2、系统向服务器发起分析数据请求。

3、显示过去一天的外汇买入年龄段分布结果，通过饼状图直观展示占比情况，占比最多的一项用红色显示，最少的一项用绿色显示，其他用随机颜色表示。点击不同颜色的板块可以在一旁显示对应的年龄段，买入总金额和所占百分比信息，默认显示最大占比年龄段信息。

**3.6、过去一天外汇买入地点分布情况分析**

1、初始化离线数据分析可视化界面

2、系统向服务器发起分析数据请求。

3、显示过去一天的外汇买入地点分布结果，与 ATM 取款金额显示在一张图上，灰色显示，查看过去一天中国境内的银行统计的外汇买入地点分布情况，结合当地外汇买入的总金额情况，由圆圈显示在地图上：交易金额总量越大圆圈越大。

**4、监控告警模块**

|  |
| --- |
| 图片 |

**4.1、参数设置**

1.    登录完成后，提供参数设置UI接口

2.    通过UI接口进入设置参数界面

3.    在参数设置界面，首先显示缺省的、或者上次设置的服务器参数。

4.    参数设置分成ATM取款交易设置、外汇买入交易设置与模拟器设置，用户必须设置所有参数并打开模拟器开关。

5.    点击保存按钮，发送保存参数设置请求。

6.    如果保存参数设置成功，则在监控指针接口显示对应参数的实时监控信息，否则返回保存失败信息。

**4.2、展示**ATM取款实时交易量

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示ATM取款实时交易量信息

4.    若没有设置参数，则按系统默认参数显示ATM取款实时交易量信息

**4.3、展示**ATM取款实时响应时间

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示ATM取款实时响应时间信息

4.    若没有设置参数，则按系统默认参数显示ATM取款实时响应时间信息

**4.4、展示外汇买入实时交易量**

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示外汇买入实时交易量信息

4.    若没有设置参数，则按系统默认参数显示外汇买入实时交易量信息

**4.5、展示外汇买入实时**响**应时间**

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示外汇买入实时响应时间信息

4.    若没有设置参数，则按系统默认参数显示外汇买入实时响应时间信息

**4.6、**展示ATM取款实时交易趋势图

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示ATM取款实时交易趋势图信息

4.    若没有设置参数，则按系统默认参数显示ATM取款实时交易趋势图信息

**4.7、**展示外汇买入实时交易趋势图

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示外汇买入实时交易趋势图信息

4.    若没有设置参数，则按系统默认参数显示外汇买入实时交易趋势图信息

**4.8、**展示ATM取款实时响应时间趋势图

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示ATM取款实时响应时间趋势图信息

4.    若没有设置参数，则按系统默认参数显示ATM取款实时响应时间趋势图信息

**4.9、**展示外汇买入实时响应时间趋势图

1.    登录完成后，提供监控指标UI接口

2.    将参数设置完成后，通过UI接口进入监控指标界面

3.    系统根据参数设置显示外汇买入实时响应时间趋势图信息

4.    若没有设置参数，则按系统默认参数显示外汇买入实时响应时间趋势图信息

**4.10、告警信息处理**

1.    进入监控指针接口后，提供警告信息处理UI接口

2.    通过UI接口进入警告信息处理界面

3.    系统根据返回的超出预期的响应时间将显示对应告警时间、故障位置、及定位故障信息

4.    用户点击处理按钮，系统发出定位故障请求

5.    若定位成功，则返回对应故障位置、已处理故障信息

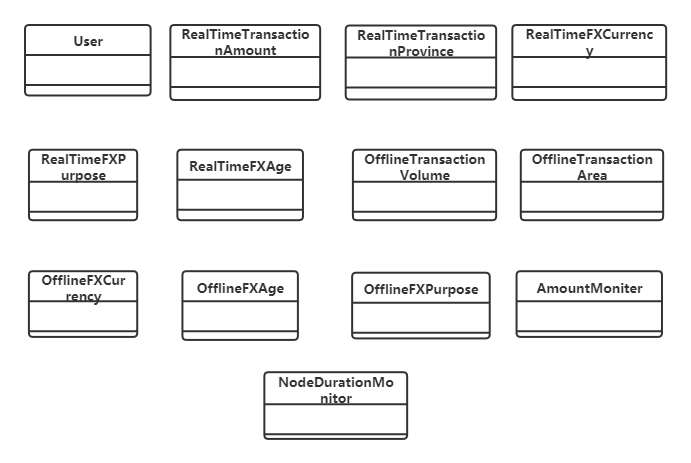
6.     若定位失败，则弹出定位失败信息

**四、模块定义**

|  |  |  |
| --- | --- | --- |
| 模块 | 组建 | 规格 |
| 账户管理模块 | 账号登陆 | 账号登陆 |
| 新建账号 | 管理员新建账号 |
| 删除账号 | 管理员删除账号 |
| 实时数据分析模块 | ATM与外汇金额实时趋势变化分析 | 银行当天每个小时内或每分钟的交易金额趋势变化折线图 |
| ATM与外汇交易地点实时分布分析 | 银行当天每个小时内或每分钟的交易地点分布变化地图 |
| 外汇买入年龄段实时分布分析 | 银行当天每个小时内或每分钟的外汇买入年龄段趋势变化饼状图 |
| 外汇买入目的实时分布分析 | 银行当天每个小时内或每分钟的外汇买入目的趋势变化饼状图 |
| 外汇买入币种实时分布分析 | 银行当天每个小时内或每分钟的外汇买入币种趋势变化饼状图 |
| 离线数据分析模块 | ATM取款金额分布离线分析 | 银行过去一天内的ATM取款金额分布条形图 |
| ATM取款地点分布离线分析 | 银行过去一天内的ATM取款地点分布图 |
| 外汇买入目的离线分析 | 银行过去一天内的外汇买入目的饼状图 |
| 外汇买入币种分布离线分析 | 银行过去一天内的外汇买入币种饼状图 |
| 外汇买入年龄段分布离线分析 | 银行过去一天内的外汇买入年龄段饼状图 |
| 外汇买入地点分布离线分析 | 银行过去一天内的外汇买入地点分布图 |
| 监控告警模块 | 参数设置 | 参数设置 |
| ATM取款实时交易量 | ATM取款实时交易量数据及实时交易趋势图 |
| ATM取款实时响应时间 | ATM取款实时响应时间数据及实时响应时间趋势图 |
| 外汇买入实时交易量 | 外汇买入实时交易量数据及实时交易趋势图 |
| 外汇买入实时响应时间 | 外汇买入实时响应时间数据及实时响应时间趋势图 |
| 告警信息处理 | 两类交易包括ATM取款、外汇买入中超出预期的响应时间的故障处理信息列表 |

**第五部分 E-R实体设计**

**一、E-R实体结构图**



**二、实体描述**

**1、用户（User）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | account | 账号 | string |
| 2 | userName | 用户名 | string |
| 3 | password | 密码 | string |
| 4 | phoneNum | 手机号 | string |
| 5 | isAdmin | 是否是管理员 | string |

**2、实时交易总金额（RealTimeTransactionAmount）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | transactionType | 交易类型 | string |
| 2 | count | 交易笔数 | string |
| 3 | volume | 交易金额总量 | string |
| 4 | aggTime | 时间戳 | string |
| 5 | amountId | id(关键字) | int |

说明：每5秒抓取一次数据

**3、实时交易地区分布**

**（RealTimeTransactionProvince）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | provinceId | id（关键字） | int |
| 2 | province | 省份 | string |
| 3 | nodeTime | 时间戳 | string |
| 4 | volume | 交易总金额 | string |
| 5 | transactionType | 交易类型 | string |

说明：每5秒抓取一次数据

**4、实时外汇币种（RealTimeFXCurrency）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | currencyID | id(关键字) | int |
| 2 | currency | 币种 | string |
| 3 | aggTime | 时间戳 | string |
| 4 | volume | 交易总金额 | string |

说明：每5秒抓取一次数据

**5、实时外汇目的（RealTimeFXPurpose）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | purposeId | id（关键字） | int |
| 2 | purpose | 买入目的 | string |
| 3 | aggTime | 时间戳 | string |
| 4 | volume | 交易总金额 | double |

说明：每5秒抓取一次数据

**6、实时外汇年龄（RealTimeFXAge）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | ageID | id(关键字) | int |
| 2 | ageRange | 年龄 | string |
| 3 | aggTime | 时间戳 | string |
| 4 | volume | 交易总金额 | string |

说明：每5秒抓取一次数据

**7、离线交易金额分布（OfflineTransactionAmount）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | 数据类型 |
| 1 | amountId | 离线总量ID | int |
| 2 | transactionType | 交易类型 | string |
| 3 | volume | 交易金额 | int |
| 4 | aggTime | 时间戳 | string |
| 5 | amountRange | 金额范围 | string |

**8、离线交易地区（OfflineTransactionArea）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | provinceId | 离线地区ID | int |
| 2 | province | 省份 | string |
| 3 | aggTime | 时间戳 | string |
| 4 | volume | 交易总金额 | double |
| 5 | transactionType | 交易类型 | string |

**9、离线外汇币种占比（OfflineFXCurrency）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | currencyId | 离线币种ID | int |
| 2 | currency | 币种 | string |
| 3 | aggTtime | 时间戳 | string |
| 4 | volume | 交易总金额 | double |

**10、离线外汇目的占比（OfflineFXPurpose）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | purposeId | 离线外汇目的ID | int |
| 2 | purpose | 买入目的 | string |
| 3 | aggTime | 时间戳 | string |
| 4 | transactionAmount | 交易总金额 | double |

**11、离线外汇年龄占比（OfflineFXAge）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | ageId | 离线外汇年龄ID | int |
| 2 | ageRange | 年龄 | int |
| 3 | aggTime | 时间戳 | string |
| 4 | volume | 交易总金额 | string |

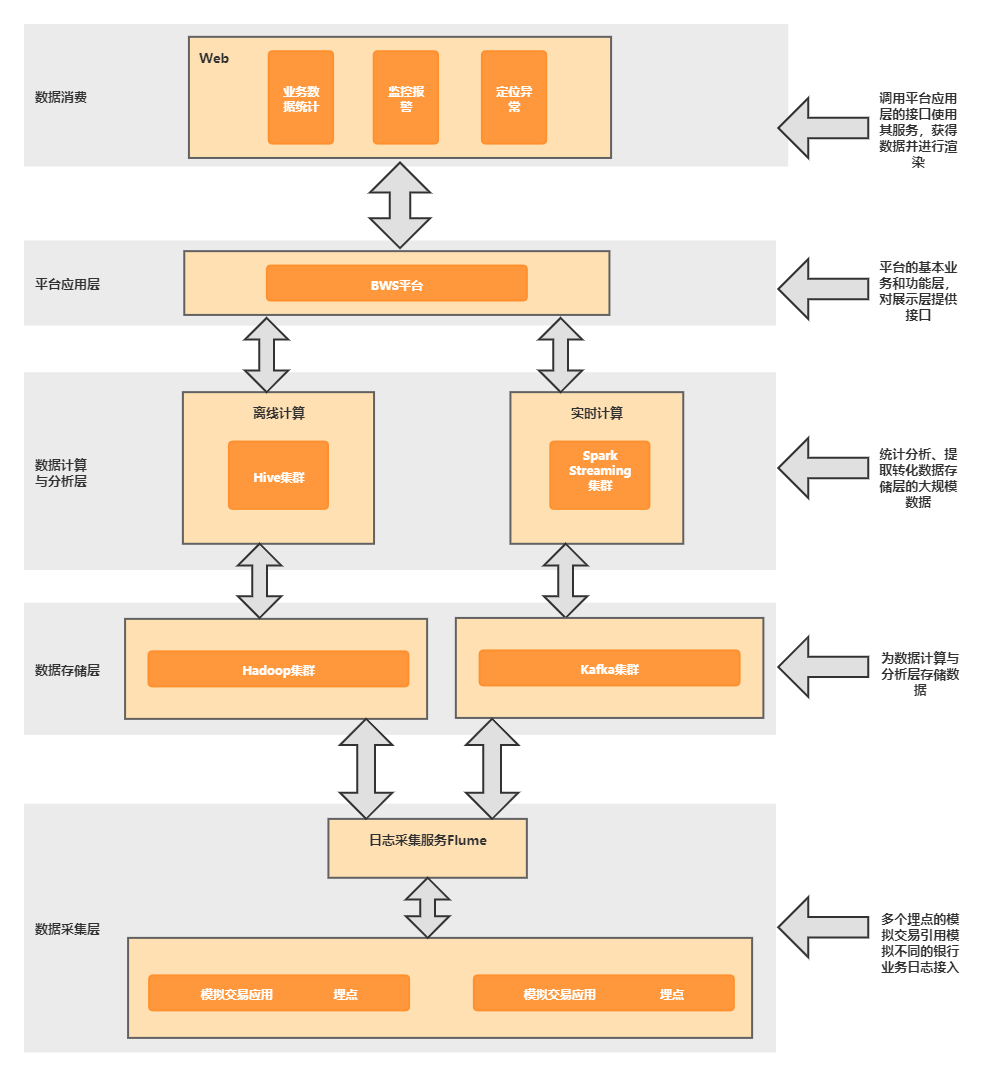
**12、响应时间监控（NodeDurationMonitor）实体描述**

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **英文名** | **中文名** | **数据类型** |
| 1 | monitorId | id | int |
| 2 | nodeDuration | 响应时间 | int |
| 3 | aggTime | 时间戳 | string |
| 4 | nodeName | 节点 | string |
| 5 | transactionType | 交易类型 | string |
| 6 | status | 节点状态 | string |

说明：每5秒写入一次数据

**第六部分 总体设计**

**一、银行监控告警平台逻辑架构设计**



**1、数据采集层**

**1.1、**Flume

|  |
| --- |
| 图片 |

数据采集层采用Flume：Flume使用JRuby来构建，所以依赖Java运行环境。Flume设计成一个分布式的管道架构，可以看作在数据源和目的地之间有一个Agent的网络，支持数据路由。

它核心是agent，这个agent对外有两个进行交互的地方，一个是接受数据的输入source，一个是数据的输出sink，sink负责将数据发送到外部指定的目的地。source接收到数据之后，将数据发送给channel，chanel作为一个数据缓冲区会临时存放这些数据，随后sink会将channel中的数据发送到指定的地方—-例如HDFS等，注意：只有在sink将channel中的数据成功发送出去之后，channel才会将临时数据进行删除，这种机制保证了数据传输的可靠性与安全性。

1.2、Hadoop

|  |
| --- |
| 图片 |

Hadoop主要的任务部署分为3个部分，分别是：Client机器，主节点和从节点。主节点主要负责Hadoop两个关键功能模块HDFS、Map Reduce的监督。当Job Tracker使用Map Reduce进行监控和调度数据的并行处理时，名称节点则负责HDFS监视和调度。每个从节点既扮演者数据节点的角色又冲当与他们主节点通信的守护进程。

1.3、Kafka

|  |
| --- |
| 图片 |

Kafka是一个分布式消息队列。Kafka对消息保存时根据Topic进行归类，发送消息者称为Producer，消息接受者称为Consumer，此外kafka集群有多个kafka实例组成，每个实例(server)称为broker。无论是kafka集群，还是consumer都依赖于zookeeper集群保存一些meta信息，来保证系统可用性。

**2、数据存储层**

2.1、Hive集群

Hive 是基于 Hadoop 的一个数据仓库工具，可以将结构化的数据文件映射为一张数据库表，并提供完整的 SQL 查询功能，将类 SQL 语句转换为 MapReduce 任务执行。

2.2、Spark集群

Spark应用程序在群集上作为独立的进程集运行，由SparkContext 主程序中的对象（称为驱动程序）协调。要在集群上运行，SparkContext可以连接到几种类型的集群管理器 （Spark自己的独立集群管理器Mesos或YARN），它们可以在应用程序之间分配资源。连接后，Spark会在集群中的节点上获取执行程序，这些节点是运行计算并为您的应用程序存储数据的进程。接下来，它将您的应用程序代码（由传递给SparkContext的JAR或Python文件定义）发送给执行者。最后，SparkContext将任务发送给执行程序以运行。

**3、数据计算与分析层**



3.1、离线分析

采用Hive数仓，通过离线etl数据分析。Hive是建立在 Hadoop 上的数据仓库基础构架。它提供了一系列的工具，可以用来进行数据提取、转化、加载（ETL Extract-Transform-Load ），这是一种可以存储、查询和分析存储在 Hadoop 中的大规模数据的机制。

3.2、实时分析

通过spark streaming实时获取数据，再进行实时etl数据分析，映射到数据库中。

**4、平台应用层**

平台的基本业务控制和基本功能层，为展示层提供接口服务。应用MVC架构，将数据与业务逻辑分离，提高代码重用率和程序的可维护性，有利于团队开发。使用SpringBoot+Mybatis技术栈实现。

**5、展示层**

通过Web展示交互界面，应用Vue框架，所有通信采用json格式。

**二、物理架构设计**

1）数据库

系统数据库部署在MySQL上，离线日志文件从HDFS经过处理之后映射到MySQL中，实时数据通过SparkStreaming处理后映射到数据库。

2）客户端

PC客户端只要求IE浏览器支持；

**三、技术架构设计**

**1、MVC模型设计**

系统采用web的MVC系统架构，实现数据，显示和处理相分离。模型(Model)负责数据管理，视图(View)负责数据显示，控制器(Controller)负责业务逻辑和响应策略。

1）Model层

封装数据源和所有基于对这些数据的操作，是实际进行数据处理的地方。Moder层向服务器请求数据，并把数据处理好交给View层进行展示

2）View层

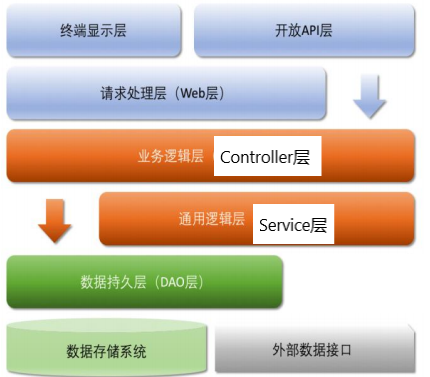
View负责数据显示与数据输入，是应用和用户之间的接口，是对数据源Model的封装显示。

3）Controller层

封装的是外界作用于模型的操作，Controller在Model和View之间起到了沟通的作用

具体我们通过SpringBoot+MyBatis来实现MVC模型：

**分层结构如图：默认上层依赖于下层，箭头关系表示可直接依赖**



• 开放 API 层：可直接封装 Service 接口暴露成 RPC 接口；通过 Web 封装成 http 接口；网关控制层等。

• 终端显示层：各个端的模板渲染并执行显示的层。当前主要是 JS 渲染，JSP 渲染等。

• Web 层：主要是对访问控制进行转发，各类基本参数校验，或者不复用的业务简单处理等。

以上层在Vue中进行设计

• Controller 层：相对具体的业务逻辑服务层，在Controller类中进行设计。

• Service 层：通用业务处理层，它有如下特征：

1） 对 Controller 层通用能力的下沉，如缓存方案、中间件通用处理。

2） 与 DAO 层交互，对多个 DAO 的组合复用。

• DAO 层：数据访问层，使用mybatis-plus与底层 MySQL等进行数据交互。

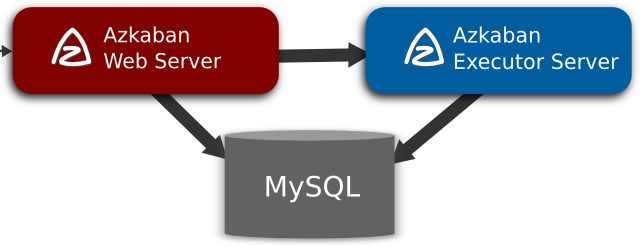
• 数据存储系统：MySQL

• 外部数据接口：外部（应用）数据存储服务提供的接口，是数据迁移场景接口。

**2、大数据模型设计**

2.1、Azkaban

Azkaban是一个批量工作流任务调度器，其架构图：

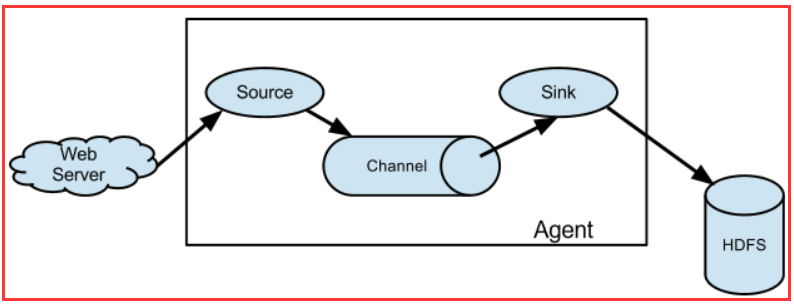


* Webserver：主要负责权限验证、项目管理、作业流下发等工作；
* Executor：主要负责作业流/作业的具体执行以及搜集执行日志等工作；
* MySQL：用于存储作业/作业流的执行状态信息。

2.2、Flume

flume的数据流由事件(Event)贯穿始终。

其体系结构：



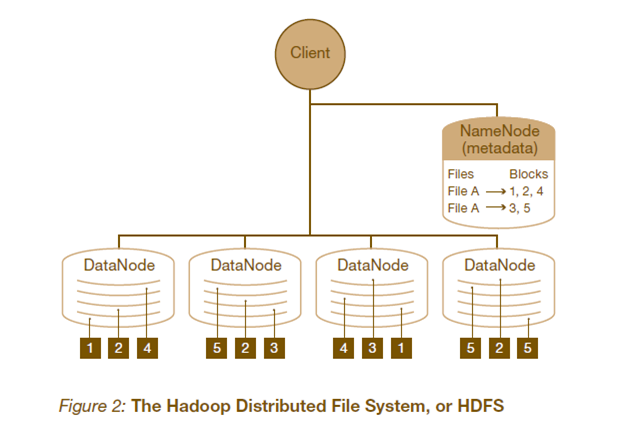
* Client：Client生产数据，运行在一个独立的线程。
* Event： 一个数据单元，消息头和消息体组成。（Events可以是日志记录、 avro 对象等。）
* Flow： Event从源点到达目的点的迁移的抽象。
* Agent： 一个独立的Flume进程，包含组件Source、 Channel、 Sink。（Agent使用JVM 运行Flume。每台机器运行一个agent，但是可以在一个agent中包含

多个sources和sinks。）

* Source： 数据收集组件。（source从Client收集数据，传递给Channel）
* Channel： 中转Event的一个临时存储，保存由Source组件传递过来的Event。（Channel连接 sources 和 sinks ）
* Sink： 从Channel中读取并移除Event， 将Event传递到FlowPipeline中的下一个Agent（如果有的话）（Sink从Channel收集数据，运行在一个独立线程。）

2.3、Hadoop

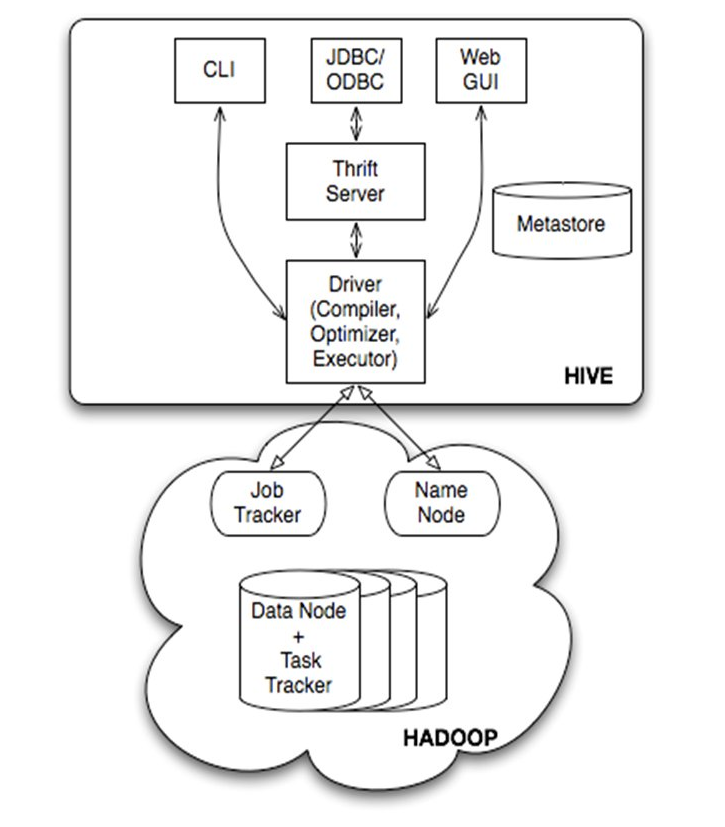
Hadoop是一个分布式系统基础架构，实现了一个分布式文件系统组件HDFS，提供了一个处理大量半结构化数据集合的编程模型MapReduce。



一个典型的HDFS集群包含一个NameNode节点和多个DataNode节点。NameNode节点负责整个HDFS文件系统中的文件的元数据的保管和管理，集群中通常只有一台机器上运行NameNode实例，DataNode节点保存文件中的数据，集群中的机器分别运行一个DataNode实例。在HDFS中，NameNode节点被称为名称节点，DataNode节点被称为数据节点。DataNode节点通过心跳机制与NameNode节点进行定时的通信。

2.4、Hive

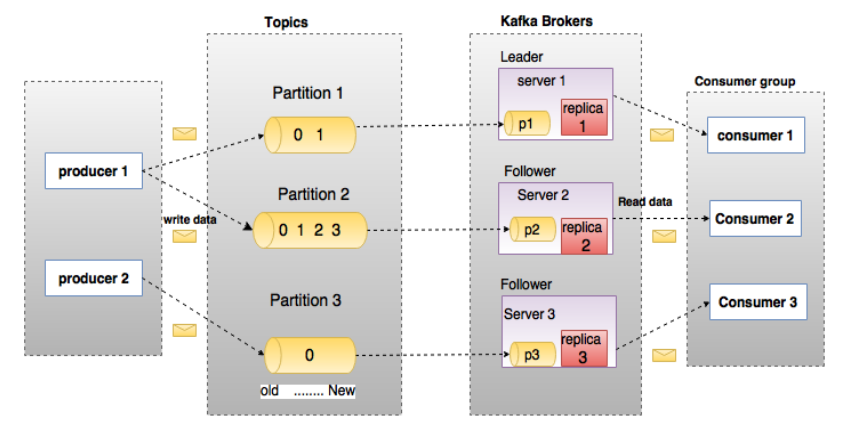
Hive是基于Hadoop的一个数据仓库工具，用来进行数据提取、转化、加载，这是一种可以存储、查询和分析存储在Hadoop中的大规模数据的机制。hive数据仓库工具能将结构化的数据文件映射为一张数据库表，并提供 SQL 查询功能，能将SQL语句转变成MapReduce任务来执行。



2.5、Kafka

Kafka是一个高性能跨语言分布式发布/订阅消息队列系统。

其结构如图所示



2.6、Spark Streaming

Spark Streaming 是Spark核心API的一个扩展，可以实现高吞吐量的、具备容错机制的实时流数据的处理。支持从多种数据源获取数据，包括Kafk、Flume、Twitter、ZeroMQ、Kinesis 以及TCP sockets，从数据源获取数据之后，可以使用诸如map、reduce、join和window等高级函数进行复杂算法的处理。最后还可以将处理结果存储到文件系统，数据库。

**3、SpringBoot**

采用SpringBoot框架，接口与接口之间、前后端之间的数据传输都使用 JSON 格式。

* @RestController 注解
* 常用数据类型转为 JSON 格式
* 使用 fastjson 封装统一返回的数据结构

SpringBoot所具备的特征有：

（1）可以创建独立的Spring应用程序，并且基于其Maven或Gradle插件，可以创建可执行的JARs和WARs；

（2）内嵌Tomcat或Jetty等Servlet容器；

（3）提供自动配置的“starter”项目对象模型（POMS）以简化Maven配置；

（4）尽可能自动配置Spring容器；

（5）提供准备好的特性，如指标、健康检查和外部化配置；

（6）不需要XML配置。

**4、Vue**

Vue.js 是一个JavaScriptMVVM库，是一套构建用户界面的渐进式框架。它是以数据驱动和组件化的思想构建的，采用自底向上增量开发的设计。相比于Angular.js，Vue.js提供了更加简洁、更易于理解的API。

**5、Mybatis**

MyBatis-Plus（简称 MP）是一个 MyBatis 的增强工具，在 MyBatis 的基础上只做增强不做改变，为简化开发、提高效率而生。MyBatis 是一款优秀的持久层框架，它支持自定义 SQL、存储过程以及高级映射。MyBatis 免除了几乎所有的 JDBC 代码以及设置参数和获取结果集的工作。MyBatis 可以通过简单的 XML 或注解来配置和映射原始类型、接口和 Java POJO（Plain Old Java Objects，普通老式 Java 对象）为数据库中的记录。

**第七部分 用户界面设计**

**一、页面布局设计**

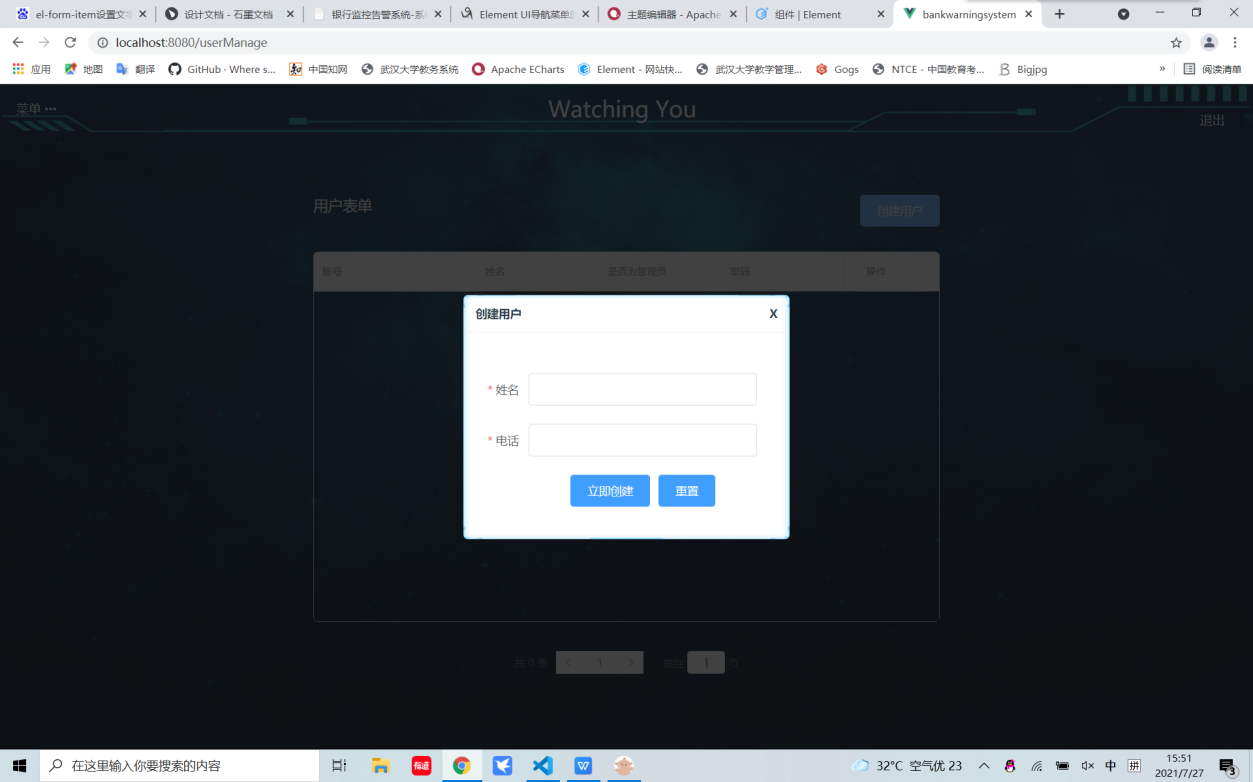
1、WEB 登录界面：

|  |
| --- |
| 图片 |

**说明：**1、输入用户账号，密码进行登，点击登录按钮，登录成功跳转页面到离线数据分析页面；

2、登录失败显示出错信息，清空文本框。

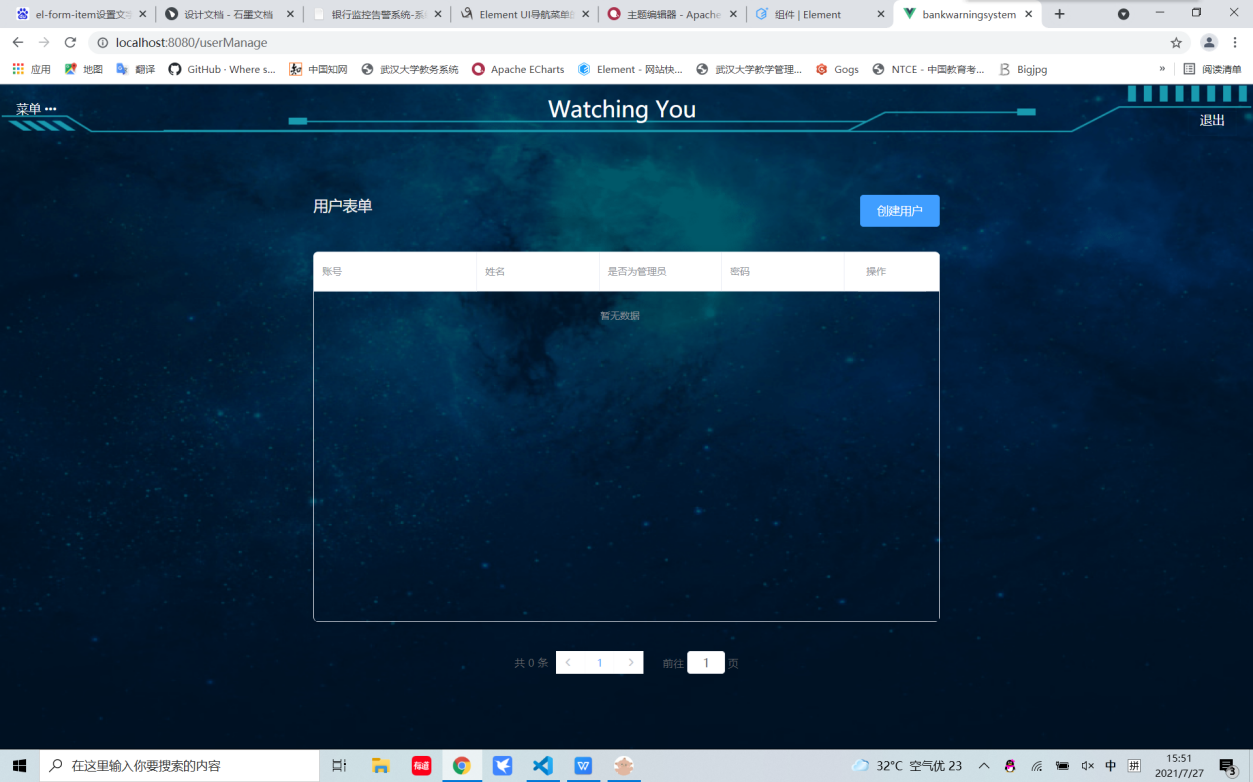
2、创建用户界面：



**说明：**1、管理员可以添加员工账号；

2、输入用户名和电话创建用户，默认密码123456；

3、用户信息界面：



**说明：**1、进入用户管理页面；

2、查看用户信息，点击删除可以移除对应账号。

4、实时业务指标分析界面：

|  |
| --- |
| 图片 |

**说明：**

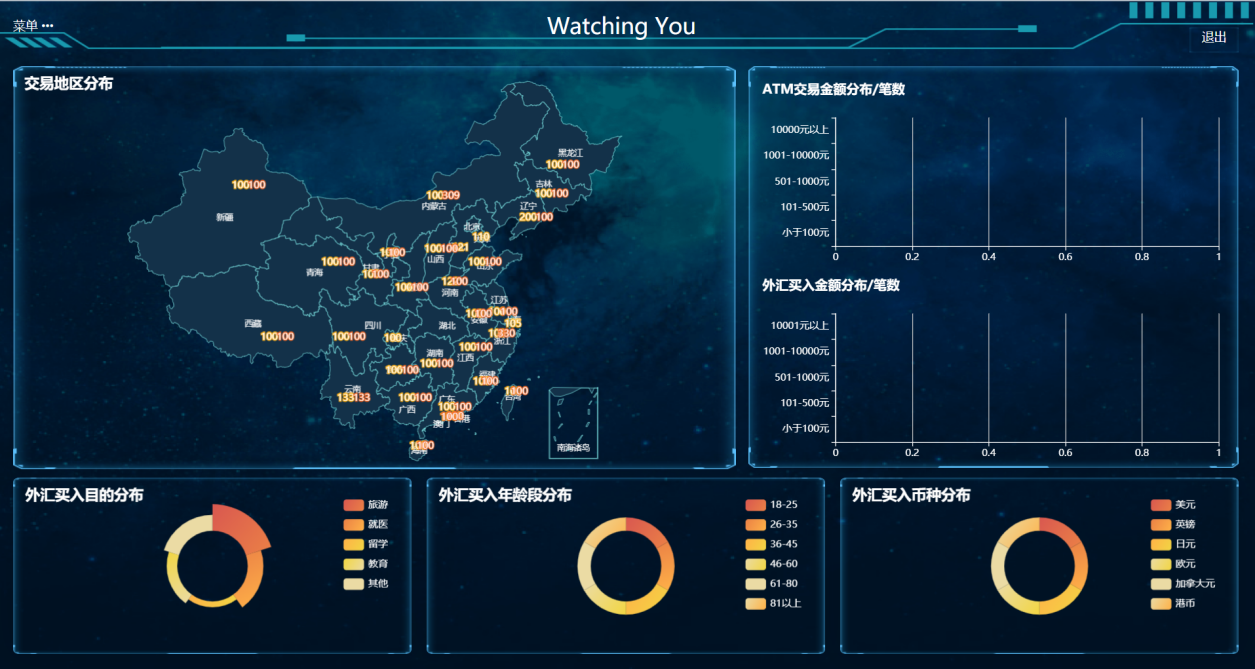
1.点击左边导航菜单栏进入实时业务指标界

2.获取数据成功则界面各板块有数据分析展示，展示实时5秒的数据。获取失败则在界面上方提示“获取数据失败“，趋势图展示最近五分钟的交易金额趋势情况

3.可以左右拉动折线图的横轴进度条查看未显示出的数据

5.界面元素有：展示ATM和外汇交易金额随时间变化的两个折线图，展示ATM和外汇交易分布地点的地图，展示外汇交易目的占比的多个环形图，展示外汇币种占比的饼图，展示外汇买入年龄占比的饼图

5、离线业务指标分析界面：



**说明：**

1、点击左边菜单栏进入离线数据分析界面；

2、查看过去一天ATM取款金额分布情况、

过去一天ATM取款地点分布情况（由黄色显示，交易金额越大圆圈越大)、

过去一天外汇买入地点分布情况（由黄色显示，交易金额越大圆圈越大）、

过去一天外汇买入目的占比（下方环形图显示，光标悬浮在各个板块上可查看富文本）、

过去一天外汇买入币种占比（下方环形图显示，光标悬浮在各个板块上可查看富文本）、

过去一天外汇买入年龄段占比（下方环形图显示，光标悬浮在各个板块上可查看富文本）。

5、监控指标页面：

|  |
| --- |
| 图片 |

**说明：**

1.    进入监控指标页面后，可以看见根据参数设置的ATM取款、外汇买入的实时交易总量和实时响应时间。

2.    当鼠标悬浮在ATM取款和外汇买入的实时交易总量趋势图的红点(因超出预期响应时间而标示)上时，会显示该时间段有多少笔交易量是超出预期响应时间，往左右拉动可以完整显示一天的实时交易总量。

3.    当鼠标悬浮在ATM取款和外汇买入的实时响应时间趋势图的红点(因超出预期响应时间而标示)上时，会显示该时间段有多少笔交易量是超出预期响应时间，往左右拉动可以完整显示一天的实时响应时间。

4.    按下警告信息处理按钮会跳转到警告信息处理页面。

6、设置参数页面：

|  |
| --- |
| 图片 |

**说明**：

1.    进入参数设置页面后，根据用户实际需求填写对应参数。

2.    填写完成后，按下模拟器开关并保存。

7、警告信息页面：

|  |
| --- |
| 图片 |

**说明：**

1.    进入警告信息处理页面后，可查看对应警告信息。

2.   点击处理按钮可以执行处理操作，再次进入该界面对应节点不会再显示，趋势图红点也消失

3. 查看告警信息节点占比情况，条形图显示，标出最大占比和最小占比节点。

**第八部分 运行环境和部署**

**一、运行环境**

**1、故障监控服务器环境**

1）Red Hat(64bit)操作系统

2）8GB 内存

3）40GB 以上的硬盘空间，不包括数据存储空间

4）JDK8

5）Mysql 8.0.19

6）Application Server：Tomcat5.5 以上版本

**2、客户机器环境**

1）Pentium III 或以上微处理器（CPU）；

2）建议使用 Windows 10 操作系统

3）内存：建议 4G 及以上

4）WEB：要求 IE7 以上版本，最好 IE8 以上版本。

5）能够连接互联网

**3、开发环境要求**

|  |  |  |
| --- | --- | --- |
| 项目 | 名称 | 版本 |
| 开发平台 | Windows | 10 |
| 开发工具 | VSCode & IDEA | VSCode 1.57.1 IDEA 2019.3.2 |
| 代码管理工具 | Azkaban | - |
| 开发环境 | JRE | 1.8.0 |

**二、系统性能要求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| # | 项目 | 模块 | 级别 | 技术参数 |
| 1 | 设计实现技术指标 | 系统架构 | A | MVC架构设计 |
| 2 | 面向对象开发语言与框架 | A | SpringBoot+Mybatis 和 VUE |
| 3 | 注释和文档 | A | 符合CMMI软件开发过程标准文档（至少提供：需求、概要、详细设计、测试报告、部署和环境、用户手册），代码注释量>=30%。 |
| 4 | 模块化和适合实训 | A | SOA设计、模块化，保证系统各模块单元较强的独立性适合实训教学。 |
| 5 | 测试覆盖率 | A | 功能覆盖率>=100%，业务覆盖率>=100%，语言覆盖率>=100%，逻辑覆盖率>=80%。 |
| 6 | 资源利用率要求 | CPU占用率 | B | <=50%利用率（附近标准配置） |
| 7 | 内存使用率 | B | <=75%利用率（附近标准配置） |
| 8 | 响应时间要求 | 服务器 | B | <=100ms（附近标准配置） |
| 9 | 网络 | B | <=100ms（附近标准配置） |
| 10 | 客户端 | B | <=5s（附近标准配置） |
| 11 | 系统稳定性要求 | 成熟性 | A | 真实的用户，成功使用本系统 |
| 12 | 稳定性 | B | 无故障运行时间>=365天，系统恢复时间<=2小时。 |
| 13 | Web服务端 | 网页部署 | A | 支持协议定制，支持安全过滤，支持消息队列，多种模式重发，支持成功检测。消息延迟<=2S，丢包<=0.001%。 |
| 14 | 集成环境部署 | PC端 | A | Microsoft Windows XP 及以上 |
| 说明：级别（A:表示非常重要必须达到的技术性能要求,B:表示重要推荐达到的技术性能要求,C：表示非重要可以弱化的技术性能要求.） | | | | |