link: null
title: 珠峰架构师成长计划
description: null
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=473 sentences=368, words=5216

## 1.安装 #

### 1.1.安装mysql #

- mysql (https://dev.mysql.com/downloads/mysql/)

### 1.2.安装redis #

- redis (https://redis.io/)

### 1.3.安装nginx #

- nginx (http://nginx.org/en/download.html)

## 2. 配置nginx #

### 2.1 nginx.conf #

```
http {
    include       mime.types;
    default_type  application/octet-stream;

    log_format  main '$time_iso8601    -    -    $remote_addr    $http_host    $status    $request_time    $request_length    $body_bytes_sent    15d04347-be16-
b9ab-0029-24e4b6645950    -    -    9689c3ea-5155-2df7-a719-e90d2dedeb2c 937ba755-116a-18e6-0735-312cba23b00c    -    -    $request_uri    $http_user_agent    -
sample=-&_UC_agent=-&device_id=-&-    -    -    -';
    access_log  logs/access.log  main;
    server {
        listen       80;
        server_name  localhost;
        if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2}):(\d{2})") {
         set $year $1;
         set $month $2;
         set $day $3;
         set $hour $4;
         set $minute $5;
        }
        access_log  logs/$year$month-$day-$hour-$minute.log  main;
        location / {
            root   html;
            index  index.html index.htm;
        }

        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
```

字段 含义 $time_iso8601 服务器时间的ISO 8610格式 $remote_addr 客户端地址 $http_host 主机名 $status HTTP响应代码 $request_time 处理客户端请求使用的时间 $request_length 请求的长度
$body_bytes_sent 传输给客户端的字节数 $request_uri 包含一些客户端请求参数的原始URI $http_user_agent 客户端用户代理

## 3. 下载项目 #

```
git clone https:
```

## 4.埋点上报 #

### 4.1 安装 #

```
cd monitor/sdk
cnpm install
```

### 4.2 配置 #

```
window.dt && dt.set({
  pid: 'template',
  uuid: 'uuid',
  ucid: 'ucid',
  is_test: true,
  record: {
    time_on_page: true,
    performance: true,
    js_error: true,

    js_error_report_config: {
      ERROR_RUNTIME: true,
      ERROR_SCRIPT: true,
      ERROR_STYLE: true,
      ERROR_IMAGE: true,
      ERROR_AUDIO: true,
      ERROR_VIDEO: true,
      ERROR_CONSOLE: true,
      ERROR_TRY_CATCH: true,

      checkErrrorNeedReport: function(desc, stack){
        return true
      }
    }
  },

  version: '1.0.0',

  getPageType: function(location){ return `${location.host}${location.pathname}` }
})
```

### 4.3 监控类型 #

#### 4.3.1 time_on_page(用户在线时长统计) #

- t_r_duration_distribution

```
const OFFLINE_MILL = 15 * 60 * 1000
const SEND_MILL = 5 * 1000

let lastTime = Date.now()
window.addEventListener('click', () => {

  const isTimeOnPageFlagOn = _.get(
    commonConfig,
    ['record', 'time_on_page'],
    _.get(DEFAULT_CONFIG, ['record', 'time_on_page'])
  )
  const isOldTimeOnPageFlagOn = _.get(commonConfig, ['online'], false)
  const needRecordTimeOnPage = isTimeOnPageFlagOn || isOldTimeOnPageFlagOn
  if (needRecordTimeOnPage === false) {
    debugLogger(`config.record.time_on_page值为false, 跳过停留时长打点`)
    return
  }

  const now = Date.now();
  const duration = now - lastTime;
  if (duration > OFFLINE_MILL) {
    lastTime = Date.now()
  } else if (duration > SEND_MILL) {
    lastTime = Date.now()
    debugLogger('发送用户留存时间埋点, 埋点内容 => ', { duration_ms: duration })

    log.product(10001, { duration_ms: duration });
  }
}, false)
```

```
{
    "d": {
        "type": "product",
        "code": 10001,
        "detail": {
            "duration_ms": 21553
        },
        "extra": {}
    }
}
```

#### 4.3.2 performance(页面载入性能) #

#### 4.3.2.1 sdk\src\index.js #

sdk\src\index.js

```
window.onload = () => {

  const isPerformanceFlagOn = _.get(
    commonConfig,
    ['record', 'performance'],
    _.get(DEFAULT_CONFIG, ['record', 'performance'])
  )
  const isOldPerformanceFlagOn = _.get(commonConfig, ['performance'], false)
  const needRecordPerformance = isPerformanceFlagOn || isOldPerformanceFlagOn
  if (needRecordPerformance === false) {
    debugLogger(`config.record.performance值为false, 跳过性能指标打点`)
    return
  }

  const performance = window.performance
  if (!performance) {

    console.log('你的浏览器不支持 performance 接口')
    return
  }
  const times = performance.timing.toJSON()

  debugLogger('发送页面性能指标数据, 上报内容 => ', {
    ...times,
    url: `${window.location.host}${window.location.pathname}`
  })

  log('perf', 20001, {
    ...times,
    url: `${window.location.host}${window.location.pathname}`
  })
}
```

```
{
    "d": {
        "type": "perf",
        "code": 20001,
        "detail": {
            "connectStart": 1592020165386,
            "navigationStart": 1592020165383,
            "loadEventEnd": 0,
            "domLoading": 1592020165401,
            "secureConnectionStart": 0,
            "fetchStart": 1592020165386,
            "domContentLoadedEventStart": 1592020165630,
            "responseStart": 1592020165393,
            "responseEnd": 1592020165394,
            "domInteractive": 1592020165630,
            "domainLookupEnd": 1592020165386,
            "redirectStart": 0,
            "requestStart": 1592020165387,
            "unloadEventEnd": 1592020165398,
            "unloadEventStart": 1592020165397,
            "domComplete": 1592020165630,
            "domainLookupStart": 1592020165386,
            "loadEventStart": 1592020165630,
            "domContentLoadedEventEnd": 1592020165630,
            "redirectEnd": 0,
            "connectEnd": 1592020165386,
            "url": "localhost:9000/"
        },
        "extra": {}
    }
}
```

#### 4.3.3 js_error(页面报错) #

错误类型 含义 ERROR_RUNTIME js运行时报错 ERROR_SCRIPT js资源加载失败 ERROR_IMAGE 图片资源加载失败 ERROR_AUDIO 音频资源加载失败 ERROR_VIDEO 视频资源加载失败 ERROR_CONSOLE vue运行时报错 ERROR_TRY_CATCH 未catch错误

##### 4.3.3.1 ERROR_RUNTIME #

```
window.addEventListener('error', function (event) {

    var errorTarget = event.target
    if (errorTarget !== window && errorTarget.nodeName && LOAD_ERROR_TYPE[errorTarget.nodeName.toUpperCase()]) {
      handleError(formatLoadError(errorTarget))
    } else {

      let { message, filename, lineno, colno, error } = event
      handleError(formatRuntimerError(message, filename, lineno, colno, error))
    }
}, true)

function formatRuntimerError (message, source, lineno, colno, error) {
  return {
    type: ERROR_RUNTIME,
    desc: message + ' at ' + source + ':' + lineno + ':' + colno,
    stack: error && error.stack ? error.stack : 'no stack'
  }
}
```

```
console.log(a.b);
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_JS_RUNTIME_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "Uncaught ReferenceError: a is not defined at http://localhost:9000/:53:21",
            "stack": "ReferenceError: a is not defined\n    at http://localhost:9000/:53:21"
        }
    }
}
```


#### 4.3.3.2 LOAD_ERROR #

```
    <script src="/error.js">script>
    <link rel="stylesheet" href="/error.css">
    <img src="/error.gif" alt="">
    <audio src="/error.mp3">audio>
    <video src="/error.mp4">video>
```

```
function formatLoadError (errorTarget) {
  return {
    type: LOAD_ERROR_TYPE[errorTarget.nodeName.toUpperCase()],
    desc: errorTarget.baseURI + '@' + (errorTarget.src || errorTarget.href),
    stack: 'no stack'
  }
}
```


```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_SCRIPT_LOAD_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "http://localhost:9000/@http://localhost:9000/error.js",
            "stack": "no stack"
        }
    }
}
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_CSS_LOAD_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "http://localhost:9000/@http://localhost:9000/error.css",
            "stack": "no stack"
        }
    }
}
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_IMAGE_LOAD_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "http://localhost:9000/@http://localhost:9000/error.gif",
            "stack": "no stack"
        }
    }
}
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_VIDEO_LOAD_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "http://localhost:9000/@http://localhost:9000/error.mp4",
            "stack": "no stack"
        }
    }
}
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_AUDIO_LOAD_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "http://localhost:9000/@http://localhost:9000/error.mp3",
            "stack": "no stack"
        }
    }
}
```

#### 4.3.3.2 LOAD_ERROR #

```
console.error('vue error');
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_CONSOLE_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "vue error"
        }
    }
}
```

#### 4.3.3.3 ERROR_TRY_CATCH #

```
function exec(){
    throw new Error('未捕获错误');
}
window.dt.tryJS.wrap(exec);
exec._wrapped();
```

```
{
    "d": {
        "type": "error",
        "code": 7,
        "detail": {
            "error_no": "页面报错_TRY_CATCH_ERROR",
            "url": "localhost:9000/"
        },
        "extra": {
            "desc": "未捕获错误",
            "stack": "Error: 未捕获错误\n    at Function.exec (http://localhost:9000/:56:17)\n    at Function.func._wrapped (webpack-internal:///./src/js-tracker/try.js:31:21)\n    at http://localhost:9000/:59:14"
        }
    }
}
```

### 4.3.4 手工上报 #

```
export const Elog = log.error = (code, detail, extra) => {
  return log('error', code, detail, extra)
}

export const Plog = log.product = (code, detail, extra) => {
  return log('product', code, detail, extra)
}

export const Ilog = log.info = (code, detail, extra) => {
  return log('info', code, detail, extra)
}
```

```
window.dt.product(19999, {}, {})
```

```
{
    "d": {
        "type": "product",
        "code": 19999,
        "detail": {},
        "extra": {}
    }
}
```

## 5.数据处理 #

### 5.1 安装server #

```
cd server
npm install
```

### 5.2 配置 #

#### 5.2.1 配置mysql #

server\src\configs\mysql.js

```
const development = {
  host: '127.0.0.1',
  port: '3306',
  user: 'root',
  password: '5f8b8a5d650637f8',
  database: 'platform'
}
```

#### 5.2.2 配置redis #

server\src\configs\redis.js

```
const development = {
  host: '127.0.0.1',
  port: '6379'
}
```

### 5.3 数据解析 #

#### 5.3.1 数据存储 #

##### 5.3.1.1 表名说明 #

- 所有表都以 t_开头
- 原始表添加 _o后缀，即 t_o_
- 结果表添加 -r后续，即 t_r_
- 表名、字段名默认使用下划线方式命名，不区分大小写
- 数据库编码字符集为 utf8mb4
- 记录ID用 unsigned bigint
- 如果字段名有关键字，需要加 _c前缀
- 所有表中必须有 update_time 和 create_time字段

##### 5.3.1.2 数据库表 #

t_o_project 项目名

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '项目id' display_name

varchar(50) NOT NULL DEFAULT '' COMMENT '项目名称' project_name

varchar(50) NOT NULL DEFAULT '' COMMENT '项目代号(替代项目id, 方便debug)' c_desc

varchar(100) NOT NULL DEFAULT '' COMMENT '备注信息' rate

int(10) NOT NULL DEFAULT '10000' COMMENT '数据抽样比率' is_delete

tinyint(1) NOT NULL DEFAULT '0' COMMENT '是否已删除(1 => 是, 0 => 否)' create_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '创建人ucid' update_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '更新人ucid' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_behavior_distribution

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' code

varchar(50) NOT NULL DEFAULT '' COMMENT '用户行为标识' name

varchar(50) NOT NULL DEFAULT '' COMMENT '用户点击名称' url

varchar(200) NOT NULL DEFAULT '' COMMENT '用户点击页面' total_count

int(10) NOT NULL DEFAULT '0' COMMENT '点击总量' count_at_time

varchar(30) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同有三种可能, hour => YYYY-MM-DD_HH, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(20) NOT NULL DEFAULT 'day' COMMENT '统计尺度(hour/day/month)' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布详情记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_duration_distribution 用户停留时间表

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' total_stay_ms

bigint(20) NOT NULL DEFAULT '0' COMMENT '总停留毫秒数' total_uv

int(10) NOT NULL DEFAULT '0' COMMENT '总uv数(从uv表中获取)' count_at_time

varchar(30) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同有两种可能, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(20) NOT NULL DEFAULT 'day' COMMENT '统计尺度(day/month)' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布详情记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_http_error_distribution

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' total_count

int(10) NOT NULL DEFAULT '0' COMMENT 'http响应码总数' http_code_2xx_count

int(10) NOT NULL DEFAULT '0' COMMENT 'http响应码2xx总数' http_code_3xx_count

int(10) NOT NULL DEFAULT '0' COMMENT 'http响应码3xx总数' http_code_4xx_count

int(10) NOT NULL DEFAULT '0' COMMENT 'http响应码4xx总数' http_code_5xx_count

int(10) NOT NULL DEFAULT '0' COMMENT 'http响应码5xx总数' http_code_other_count

int(10) NOT NULL DEFAULT '0' COMMENT '其他http响应码总数' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布详情记录id' count_at_time

varchar(30) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同有三种可能, hour => YYYY-MM-DD_HH, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(20) NOT NULL DEFAULT 'hour' COMMENT '统计尺度(hour/day/month)' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间'

t_r_page_view

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' total_count

int(10) NOT NULL DEFAULT '0' COMMENT 'pv数' count_at_time

varchar(30) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同有三种可能, hour => YYYY-MM-DD_HH, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(20) NOT NULL DEFAULT 'hour' COMMENT '统计尺度(hour/day/month)' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_system_browser

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' browser

varchar(20) NOT NULL DEFAULT '' COMMENT '浏览器品牌' browser_version

varchar(50) NOT NULL DEFAULT '' COMMENT '浏览器详情' total_count

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '数量总和' count_at_month

varchar(15) NOT NULL DEFAULT '' COMMENT '统计时间段, YYYY-MM格式 => 2018-09' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_system_runtime_version

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' runtime_version

varchar(50) NOT NULL DEFAULT '' COMMENT '应用版本号' total_count

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '数量总和' count_at_month

varchar(15) NOT NULL DEFAULT '' COMMENT '统计时间段, YYYY-MM格式 => 2018-09' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_system_device

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' device_vendor

varchar(50) NOT NULL DEFAULT '' COMMENT '设备制造商' device_model

varchar(50) NOT NULL DEFAULT '' COMMENT '设备详情' total_count

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '数量总和' count_at_month

varchar(15) NOT NULL DEFAULT '' COMMENT '统计时间段, YYYY-MM格式 => 2018-09' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_system_os

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' os

varchar(50) NOT NULL DEFAULT '' COMMENT '操作系统名' os_version

varchar(50) NOT NULL DEFAULT '' COMMENT '操作系统版本' total_count

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '数量总和' count_at_month

varchar(15) NOT NULL DEFAULT '' COMMENT '统计时间段, YYYY-MM格式 => 2018-09' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_unique_view

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' total_count

int(10) NOT NULL DEFAULT '0' COMMENT 'uv数' count_at_time

varchar(30) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同有三种可能, hour => YYYY-MM-DD_HH, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(20) NOT NULL DEFAULT 'hour' COMMENT '统计尺度(hour/day/month)' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_o_alarm_config

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT project_id

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '要报警项目id' owner_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '项目所有人id' error_type

varchar(20) NOT NULL DEFAULT '' COMMENT '报警错误类型' error_name

varchar(255) NOT NULL DEFAULT '' COMMENT '要报警错误名字' time_range_s

int(20) unsigned NOT NULL DEFAULT '0' COMMENT '报警时间范围_秒' max_error_count

int(20) unsigned NOT NULL DEFAULT '0' COMMENT '报警错误数阈值' alarm_interval_s

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '报警时间间隔_秒' is_enable

tinyint(1) unsigned NOT NULL DEFAULT '1' COMMENT '是否开启本条报警配置1：是0：否' note

varchar(255) NOT NULL DEFAULT '' COMMENT '配置说明' is_delete

tinyint(1) unsigned NOT NULL DEFAULT '0' COMMENT '是否删除(1 => 是, 0 => 否)' create_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '创建此记录的人' update_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '更新此记录的人' create_time

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '创建此记录的时间' update_time

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '更新此记录的时间'

t_o_user

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' ucid

varchar(50) NOT NULL DEFAULT '' COMMENT '贝壳ucid' account

varchar(50) NOT NULL DEFAULT '' COMMENT '账户名,不能重复' email

varchar(50) NOT NULL DEFAULT '' COMMENT '邮箱' password_md5

varchar(32) NOT NULL DEFAULT '' COMMENT 'md5后的password' nickname

varchar(20) NOT NULL DEFAULT '' COMMENT '昵称' role

varchar(50) NOT NULL DEFAULT 'dev' COMMENT '角色(dev => 开发者,admin => 管理员)' register_type

varchar(20) NOT NULL DEFAULT 'site' COMMENT '注册类型(site => 网站注册, third => 第三方登录)' avatar_url

varchar(200) NOT NULL DEFAULT '
http://ww1.sinaimg.cn/large/00749HCsly1fwofq2t1kaj30qn0qnaai.jpg' (http://ww1.sinaimg.cn/large/00749HCsly1fwofq2t1kaj30qn0qnaai.jpg')

COMMENT '头像地址, 默认使用贝壳logo' mobile

varchar(20) NOT NULL DEFAULT '' COMMENT '手机号' is_delete

tinyint(1) unsigned NOT NULL DEFAULT '0' COMMENT '是否删除(1 => 是, 0 => 否)' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_o_project_member

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT project_id

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '项目id' ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '项目参与者ucid' role

varchar(20) NOT NULL DEFAULT '' COMMENT '参与者角色(owner => 组长, dev => 成员)' need_alarm

tinyint(1) unsigned NOT NULL DEFAULT '0' COMMENT '是否需要报警(0 => 不需要, 1 => 需要)' is_delete

tinyint(1) unsigned NOT NULL DEFAULT '0' COMMENT '是否已删除(0 => 未删除, 1 => 已删除)' create_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '创建者ucid' update_ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '更新者ucid' create_time

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '创建此记录的时间' update_time

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '更新此记录的时间'

t_r_new_user_summary

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '项目id' total_count

int(10) NOT NULL DEFAULT '0' COMMENT '数量总和' count_at_time

varchar(20) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同三两种可能, hour => YYYY-MM-DD_HH, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(10) NOT NULL DEFAULT 'day' COMMENT '统计尺度(hour/day/month)' city_distribute_id

bigint(10) NOT NULL DEFAULT 0 COMMENT '城市分布详情记录id' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '更新时间'

t_r_alarm_log

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' project_id

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '要报警项目id' config_id

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '报警配置id' send_at

bigint(20) NOT NULL DEFAULT '0' COMMENT '报警时间戳' error_type

varchar(20) NOT NULL DEFAULT '' COMMENT '错误类型' error_name

varchar(255) NOT NULL DEFAULT '' COMMENT '错误名字' message

varchar(500) NOT NULL DEFAULT '' COMMENT '报警信息' create_time

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '创建时间' update_time

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '更新时间'

t_o_monitor_1_202005

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' error_type

varchar(20) NOT NULL DEFAULT '' COMMENT '异常类型(目前是四种,分别对应前端的四种报错类型: api_data =>前端数据结构报警, start_process => 启动过程异常, load_wv => Url加载空服务报警, api_code => 请求接口异常报警)' error_name

varchar(255) NOT NULL DEFAULT '' COMMENT '错误名/错误代码,用于细分错误类别' http_code

int(10) NOT NULL DEFAULT '0' COMMENT 'http状态码, 没有则为0' monitor_ext_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '详情id' during_ms

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '接口请求时长, 单位毫秒' request_size_b

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '接口请求体积, 单位b' response_size_b

int(10) unsigned NOT NULL DEFAULT '0' COMMENT '接口响应体积, 单位b' url

varchar(255) NOT NULL DEFAULT '' COMMENT '发生异常的url' country

varchar(10) NOT NULL DEFAULT '' COMMENT '所属国家' province

varchar(15) NOT NULL DEFAULT '' COMMENT '所属省份' city

varchar(15) NOT NULL DEFAULT '' COMMENT '所属城市' log_at

bigint(20) NOT NULL DEFAULT '0' COMMENT '日志记录时间' md5

char(32) NOT NULL DEFAULT '' COMMENT '记录生成MD5' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_o_monitor_ext_1_202005

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' ext_json

text COMMENT '异常记录扩展信息' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_o_uv_record_1_202005

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '项目id' uuid

varchar(50) NOT NULL DEFAULT '' COMMENT '设备唯一标识' country

varchar(10) NOT NULL DEFAULT '' COMMENT '所属国家' province

varchar(15) NOT NULL DEFAULT '' COMMENT '所属省份' city

varchar(15) NOT NULL DEFAULT '' COMMENT '所属城市' visit_at_hour

varchar(20) NOT NULL DEFAULT '' COMMENT '访问日期, 数据格式为 YYYY-MM-DD_HH demo => 2018-08-02_23' pv_count

int(10) NOT NULL DEFAULT '0' COMMENT '时间段内同一uuid访问次数, 用于计算总pv' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_city_distribution_1_202005

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' city_distribute_json

text COMMENT '扩展字段' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_r_performance_1_202005

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' sum_indicator_value

bigInt(10) NOT NULL DEFAULT '0' COMMENT '性能指标求和' pv

bigInt(10) NOT NULL DEFAULT '0' COMMENT '性能指标pv数,用于计算平均时长' indicator

varchar(50) NOT NULL DEFAULT '' COMMENT '性能指标:DNS响应时间/TCP时间/404数量/etc' url

varchar(255) NOT NULL DEFAULT '' COMMENT '页面url' city_distribute_id

bigint(20) NOT NULL DEFAULT '0' COMMENT '城市分布详情记录id' count_at_time

varchar(20) NOT NULL DEFAULT '' COMMENT '统计日期,格式根据统计尺度不同有四种可能, minute => YYYY-MM-DD_HH:mm, hour => YYYY-MM-DD_HH, day => YYYY-MM-DD, month => YYYY-MM' count_type

varchar(10) NOT NULL DEFAULT 'minute' COMMENT '统计尺度(minute/hour/day/month)' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '更新时间'

t_o_system_collection_1

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' uuid

varchar(50) NOT NULL DEFAULT '' COMMENT '设备唯一标识' browser

varchar(50) NOT NULL DEFAULT '' COMMENT '浏览器品牌' browser_version

varchar(100) NOT NULL DEFAULT '' COMMENT '浏览器版本详情' engine

varchar(100) NOT NULL DEFAULT '' COMMENT '内核名称' engine_version

varchar(100) NOT NULL DEFAULT '' COMMENT '内核版本详情' device_vendor

varchar(100) NOT NULL DEFAULT '' COMMENT '手机品牌' device_model

varchar(100) NOT NULL DEFAULT '' COMMENT '手机型号' os

varchar(50) NOT NULL DEFAULT '' COMMENT '操作系统' os_version

varchar(50) NOT NULL DEFAULT '' COMMENT '操作系统详情' country

varchar(10) NOT NULL DEFAULT '' COMMENT '所属国家' province

varchar(15) NOT NULL DEFAULT '' COMMENT '所属省份' city

varchar(15) NOT NULL DEFAULT '' COMMENT '所属城市' runtime_version

varchar(50) NOT NULL DEFAULT '' COMMENT '应用版本' visit_at_month

varchar(20) NOT NULL DEFAULT '' COMMENT '访问日期, 数据格式为 YYYY-MM demo => 2018-09' log_at

bigint(20) NOT NULL DEFAULT '0' COMMENT '日志记录时间' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '数据库更新时间'

t_o_user_first_login_at_1

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' ucid

varchar(20) NOT NULL DEFAULT '' COMMENT '用户id' first_visit_at

bigint(20) NOT NULL DEFAULT '0' COMMENT '用户首次访问时间' country

varchar(10) NOT NULL DEFAULT '' COMMENT '所属国家' province

varchar(15) NOT NULL DEFAULT '' COMMENT '所属省份' city

varchar(15) NOT NULL DEFAULT '' COMMENT '所属城市' create_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '创建时间' update_time

bigint(20) NOT NULL DEFAULT '0' COMMENT '更新时间'

t_r_error_summary_1_202005

字段 备注 id

bigint(20) unsigned NOT NULL AUTO_INCREMENT COMMENT '记录id' error_type

varchar(20) NOT NULL DEFAULT '' COMMENT '错误类型' error_name

varchar(255) NOT NULL DEFAULT '' COMMENT '错误名字' url_path

varchar(255) NOT NULL DEFAULT '' COMMENT '错误URL_PATH' city_distribution_id

bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '城市分布详情id' count_at_time

varchar(20) NOT NULL DEFAULT '' COMMENT '统计时间(按类型不同,day:YYYY-MM-DD;hour:YYYY-MM-DD_HH;minute:YYYY-MM-DD_HH:mm)' count_type

varchar(10) NOT NULL DEFAULT 'day' COMMENT '统计类型(day天,hour小时,minute分)' error_count

int(10) NOT NULL DEFAULT '0' COMMENT '数量总和' create_time

int(20) NOT NULL DEFAULT '0' COMMENT '创建时间' update_time

int(20) NOT NULL DEFAULT '0' COMMENT '更新时间'

### 5.3.2 指令 #

#### 5.3.2.1 指令帮助 #

```
node dist/fee.js -h
Usage:
  command [arguments] [options]

Global Options:
  --env                           Set NODE_ENV before running the commands
  --no-ansi                       Disable colored output

Available Commands:
 Command
  Command:Demo                    解析nginx日志, 分析pv
 CreateCache
  CreateCache:UpdatePerOneMinute  [每10分钟执行一次] 主动调用方法, 更新Redis缓存, 每10分钟更新一次
 Parse
  Parse:Device                    [按天] 解析nginx日志, 分析指定时间范围Device
  Parse:MenuClick                 [按天] 解析nginx日志, 用户点击情况
  Parse:Monitor                   [按分钟] 解析nginx日志, 分析Monitor
  Parse:Performance               [按小时] 解析nginx日志, 分析分钟级别的指定时间范围内的性能指标
  Parse:TimeOnSiteByHour          [按小时] 解析nginx日志, 分析记录指定时间范围内用户停留时长
  Parse:UV                        [按小时] 解析nginx日志, 分析记录指定时间范围内的uv
  Parse:UserFirstLoginAt          [按天] 解析nginx日志, 记录用户首次登陆时间
 SaveLog
  SaveLog:Nginx                   每一分钟读取Nginx日志文件，并解析
 Summary
  Summary:Error                   [按分钟/按小时/按天] 根据历史数据, 汇总分析错误数
  Summary:HttpError               [按天/按月] 基于数据表做统计, 统计http error分布情况
  Summary:NewUser                 [按小时/按天/按月] 根据历史数据, 汇总分析记录指定时间范围内的新增用户数
  Summary:Performance             [按小时/按天/按月] 根据历史数据, 汇总分析记录指定时间范围内的性能指标数据
  Summary:SystemBrowser           [按月] 基于数据库统计浏览器占比
  Summary:SystemDevice            [按月] 基于数据库统计设备占比
  Summary:SystemOS                [按月]基于数据库统计操作系统占比
  Summary:SystemRuntimeVersion    [按月] 基于数据库统计浏览器占比
  Summary:TimeOnSite              [按天/按月] 根据历史数据, 汇总分析记录指定时间范围内用户停留时长
  Summary:UV                      [按小时/按天/按月] 根据历史数据, 汇总分析记录指定时间范围内的uv
 Task
  Task:Manager                    任务调度主进程, 只能启动一次
 Utils
  Utils:CleanOldLog               只保留当前月内数据, 每月20号之后自动删除上个月数据
  Utils:GenerateSQL               生成项目在指定日期范围内的建表SQL
  Utils:TemplateSQL               生成项目在指定日期范围内的建表SQL
  Utils:Test                      专业粘贴调试代码
  Utils:TestUC                    测试UC接口
 WatchDog
  WatchDog:Alarm                  [根据报警配置] 监测每一条报警配置对应的项目错误
  WatchDog:Saas                   [按分钟] 检查最近5分钟内错误数是否超出阈值, 自动报警
```

#### 5.3.2.2 初始化数据库 #

- 创建 platform数据库
- npm run watch
- node dist/fee.js Utils:GenerateSQL 1 '2020-06' '2020-06' > init.sql

```
REPLACE INTO `t_o_project` (`id`, `rate`, `display_name`, `project_name`, `c_desc`, `is_delete`, `create_ucid`, `update_ucid`, `create_time`, `update_time`)
VALUES (1, 100, '示例项目', 'template', '负责人', 0, '', '', 0, 0);
```

#### 5.3.2.3 SaveLog:Nginx #

- 每一分钟读取Nginx日志文件，并解析

server\src\library\nginx\index.js

```
let logPath = appConfig.absoluteLogPath
```

server\src\configs\app.js

```
const development = {
  name: 'fee监控平台开发环境',
  port: 3000,
  proxy: false,
  absoluteLogPath: path.resolve(__dirname, '../../', 'log')
}
```

```
node dist/fee.js SaveLog:Nginx
```

```
收到数据, 当前共记录1/1条数据
清洗后的日志路径 C:\afirst\monitor\server\log\nginx\raw\month_202006\day_13\16\14.log

清洗后的日志路径 C:\afirst\monitor\server\log\nginx\json\month_202006\day_13\16\14.log
```

```
{
    "type": "perf",
    "code": 20001,
    "detail": {
        "connectStart": 1592036791666,
        "navigationStart": 1592036791361,
        "loadEventEnd": 0,
        "domLoading": 1592036791677,
        "secureConnectionStart": 0,
        "fetchStart": 1592036791362,
        "domContentLoadedEventStart": 1592036791976,
        "responseStart": 1592036791669,
        "responseEnd": 1592036791670,
        "domInteractive": 1592036791976,
        "domainLookupEnd": 1592036791365,
        "redirectStart": 0,
        "requestStart": 1592036791667,
        "unloadEventEnd": 1592036791674,
        "unloadEventStart": 1592036791674,
        "domComplete": 1592036791976,
        "domainLookupStart": 1592036791365,
        "loadEventStart": 1592036791976,
        "domContentLoadedEventEnd": 1592036791976,
        "redirectEnd": 0,
        "connectEnd": 1592036791667,
        "url": "localhost:9000/"
    },
    "extra": {},
    "common": {
        "pid": "template",
        "uuid": "uuid2",
        "ucid": "ucid",
        "is_test": true,
        "record": {
            "time_on_page": true,
            "performance": true,
            "js_error": true,
            "js_error_report_config": {
                "ERROR_RUNTIME": true,
                "ERROR_SCRIPT": true,
                "ERROR_STYLE": true,
                "ERROR_IMAGE": true,
                "ERROR_AUDIO": true,
                "ERROR_VIDEO": true,
                "ERROR_CONSOLE": true,
                "ERROR_TRY_CATCH": true
            }
        },
        "version": "1.0.0",
        "test": "b47ca710747e96f1c523ebab8022c19e9abaa56b",
        "timestamp": 1592036791977,
        "runtime_version": "1.0.0",
        "sdk_version": "1.0.40",
        "page_type": "localhost:9000/"
    },
    "md5": "382eecc357bb7c6629e139eb7eb6509e",
    "project_id": 1,
    "project_name": "template",
    "time": 1592036792,
    "ua": {
        "ua": "Mozilla/5.0 (iPhone; CPU iPhone OS 13_2_3 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/13.0.3 Mobile/15E148 Safari/604.1",
        "browser": {
            "name": "Mobile Safari",
            "version": "13.0.3",
            "major": "13"
        },
        "engine": {
            "name": "WebKit",
            "version": "605.1.15"
        },
        "os": {
            "name": "iOS",
            "version": "13.2.3"
        },
        "device": {
            "vendor": "Apple",
            "model": "iPhone",
            "type": "mobile"
        },
        "cpu": {}
    },
    "ip": "59.109.146.215",
    "country": "中国",
    "province": "北京",
    "city": "北京"
}
```


#### 5.3.2.4 Parse:Performance #

- **[按小时]** 解析nginx日志, 分析分钟级别的指定时间范围内的性能指标

```
node dist/fee.js Parse:Performance 2020-06-13_16:26  2020-06-13_16:26
```

```
{
  [项目ID,url,指标,汇总的时间]:{
    [国家,省,市],value: {指标和:614,pv和:1}
  }
}
```


```
{
  [1,"localhost:9000/","tcp_connect_ms","2020-06-13_16:26"]:{
    ["中国","北京","北京"],value: {"sum_indicator_value":614,"pv":1}
  }
}
```

#### 5.3.2.5 Summary:Performance #

- [按小时/按天/按月] 根据历史数据, 汇总分析记录指定时间范围内的性能指标数据

```
node dist/fee.js Summary:Performance "2020-06-13 16" hour
node dist/fee.js Summary:Performance 2020-06-13 day
node dist/fee.js Summary:Performance 2020-06 month
```

#### 5.3.2.5 Parse:TimeOnSiteByHour #

- [按小时] 解析nginx日志, 分析记录指定时间范围内用户停留时长
- t_r_city_distribution_1_202006
- t_r_duration_distribution

```
node dist/fee.js SaveLog:Nginx
node dist/fee.js Parse:TimeOnSiteByHour "2020-06-13 22:30"  "2020-06-13 22:30"
```

#### 5.3.2.6 Summary:TimeOnSite #

- [按小时] 解析nginx日志, 分析记录指定时间范围内用户停留时长
- t_r_duration_distribution

```
node dist/fee.js Summary:TimeOnSite "2020-06 13"  day
node dist/fee.js Summary:TimeOnSite "2020-06"  month
```

#### 5.3.2.7 Parse:Monitor #

- [按分钟] 解析nginx日志, 分析Monitor
- t_o_monitor_1_202006
- t_o_monitor_ext_1_202006

```
node dist/fee.js Parse:Monitor "2020-06-13 22:35"  "2020-06-13 22:35"
```

#### 5.3.2.8 Summary:Error #

- [按分钟/按小时/按天] 根据历史数据, 汇总分析错误数
- t_r_error_summary_1_202006

```
node dist/fee.js Summary:Error "2020-06-13 22:35"  minute
node dist/fee.js Summary:Error "2020-06-13 22" hour
node dist/fee.js Summary:Error "2020-06-13" day
```

#### 5.3.2.9 Parse:Device #

- [按天] 解析nginx日志, 分析指定时间范围Device
- t_o_system_collection_1

```
node dist/fee.js Parse:Device "2020-06-13 22:58"  "2020-06-13 22:58"
```

```
projectMap={
  1:{
    "2020-06":{
      "uuid":{
        "browser":"Mobile Safari"
      }
    }
  }
}
```

#### 5.3.2.10 Summary #

- Summary:SystemBrowser [按月] 基于数据库统计浏览器占比
- Summary:SystemDevice [按月] 基于数据库统计设备占比
- Summary:SystemOS [按月]基于数据库统计操作系统占比
- Summary:SystemRuntimeVersion [按月] 基于数据库统计浏览器占比
- t_r_system_browser
- t_r_system_device
- t_r_system_os
- t_r_system_runtime_version

```
node dist/fee.js Summary:SystemBrowser "2020-06"  month
node dist/fee.js Summary:SystemDevice "2020-06"  month
node dist/fee.js Summary:SystemOS "2020-06"  month
node dist/fee.js Summary:SystemRuntimeVersion "2020-06"  month
```

## 6.计划任务 #

- server\src\commands\task\manage.js

### 6.1 任务执行周期 #

```
1.  每分钟一次(准实时)
    1.  原始数据入库
        1.  错误数据入库(延迟2分钟)
    2.  按分钟统计
        1.  错误数据统计(延迟2分钟)
2.  每10分钟一次
    1.  原始数据入库
        1.  uv
        2.  页面性能指标
        3.  用户停留时长
    2.  按小时统计
        1.  uv
        2.  新用户数
        3.  页面性能指标
        4.  错误数据统计
3.  每小时一次
    1.  原始数据入库
        1.  设备数据
        2.  用户点击
        3.  首次登陆用户
    2.  按天统计(当天)
        1.  uv
        2.  新用户数
        3.  页面性能指标
        4.  错误数据统计
        5.  用户停留时长
4.  每六小时一次
    1.  按天统计(昨日)
        1.  uv
        2.  新用户数
        3.  页面性能指标
        4.  错误数据统计
        5.  用户停留时长
    2.  按月统计
        1.  uv
        2.  新用户数
        3.  页面性能指标
        4.  错误数据统计
        5.  用户停留时长
        6.  操作系统分布
        7.  设备分布
        8.  浏览器分布
```

### 6.2 执行任务 #

- cron (https://cron.qqe2.com/)

```
node dist/fee.js Task:saveLog
```

```
async handle (args, options) {
    let that = this
    schedule.scheduleJob('0 */1 * * * * *', function () {
      that.log('registerTaskRepeatPerlMinute 开始执行')
      that.execCommand('SaveLog:Nginx', []);
    })
  }
```

```
*  *  *  *  *  *
┬ ┬ ┬ ┬ ┬ ┬
│ │ │ │ │  |
│ │ │ │ │ └ day of week (0 - 7) (0 or 7 is Sun)
│ │ │ │ └─── month (1 - 12)
│ │ │ └────── day of month (1 - 31)
│ │ └──────── hour (0 - 23)
│ └────────── minute (0 - 59)
└──────────── second (0 - 59, OPTIONAL)
```

## 7.前台展示 #

### 7.1 启动接口 #

```
node dist/app.js
```

```
需要登录,也需要检验项目权限(Method: get) =>/api/behavior/menu
需要登录,也需要检验项目权限(Method: get) =>/api/behavior/online
需要登录,也需要检验项目权限(Method: get) =>/api/os
需要登录,也需要检验项目权限(Method: get) =>/api/browser/list
需要登录,也需要检验项目权限(Method: get) =>/api/browser
需要登录,也需要检验项目权限(Method: get) =>/api/browser/
需要登录,也需要检验项目权限(Method: get) =>/api/runtimeVersion
需要登录,也需要检验项目权限(Method: get) =>/api/error/viser/area/
需要登录,也需要检验项目权限(Method: get) =>/api/error/log/list
需要登录,也需要检验项目权限(Method: get) =>/api/error/distribution/url
需要登录,也需要检验项目权限(Method: get) =>/api/error/distribution/
需要登录,也需要检验项目权限(Method: get) =>/api/error/distribution/
需要登录,也需要检验项目权限(Method: get) =>/api/error/distribution/
不需要登录(Method: get) =>/api/log/content/test
需要登录,也需要检验项目权限(Method: post) =>/api/alarm/config/add
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/config/query
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/config/list
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/config/delete
需要登录,也需要检验项目权限(Method: post) =>/api/alarm/config/update
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/config/error_name_list
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/config/error_type_list
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/log
需要登录,也需要检验项目权限(Method: get) =>/api/alarm/log/line
需要登录,但不需要检验项目权限(Method: get) =>/api/user/detail
需要登录,但不需要检验项目权限(Method: get) =>/api/user/search
需要登录,但不需要检验项目权限(Method: post) =>/api/user/update
需要登录,但不需要检验项目权限(Method: get) =>/api/user/delete
需要登录,但不需要检验项目权限(Method: get) =>/api/user/search_uc
不需要登录(Method: post) =>/api/user/register
需要登录,但不需要检验项目权限(Method: post) =>/api/user/modify/password
需要登录,但不需要检验项目权限(Method: post) =>/api/user/modify/msg
需要登录,但不需要检验项目权限(Method: get) =>/api/user/destroy
不需要登录(Method: post) =>/api/login/site
不需要登录(Method: post) =>/api/login/uc
不需要登录(Method: get) =>/api/logout
不需要登录(Method: post) =>/api/login/normal
不需要登录(Method: get) =>/api/login/type
不需要登录(Method: post) =>/api/login
需要登录,但不需要检验项目权限(Method: get) =>/api/project/item/detail
需要登录,但不需要检验项目权限(Method: post) =>/api/project/item/add
需要登录,但不需要检验项目权限(Method: get) =>/api/project/item/list
需要登录,但不需要检验项目权限(Method: get) =>/api/project/item/delete
需要登录,但不需要检验项目权限(Method: post) =>/api/project/item/update
需要登录,也需要检验项目权限(Method: post) =>/api/project/member/add
需要登录,也需要检验项目权限(Method: get) =>/api/project/member/list
需要登录,也需要检验项目权限(Method: get) =>/api/project/member/delete
需要登录,也需要检验项目权限(Method: post) =>/api/project/member/update
需要登录,也需要检验项目权限(Method: get) =>/api/project/summary/new_user/distribution_line
需要登录,也需要检验项目权限(Method: get) =>/api/project/summary/new_user/distribution_map
需要登录,也需要检验项目权限(Method: get) =>/api/performance/url_list
需要登录,也需要检验项目权限(Method: get) =>/api/performance/url/overview
需要登录,也需要检验项目权限(Method: get) =>/api/performance/project/overview
需要登录,也需要检验项目权限(Method: get) =>/api/performance/url/line_chart
需要登录,也需要检验项目权限(Method: get) =>/api/uv/count
```

### 7.2 启动前台 #

```
cd client
cnpm i
npm run start
```

### 7.3 前端应用 #

client\src\router\index.js

```
const token = getToken();
if (!token && to.name !== LOGIN_PAGE_NAME) {

    next({
      name: LOGIN_PAGE_NAME
    })
}
```

server\src\routes\api\user\index.js

```
const register = RouterConfigBuilder.routerConfigBuilder('/api/user/register', RouterConfigBuilder.METHOD_TYPE_POST, async (req, res) => {
  const body = _.get(req, ['body'], {})
}
```

server\src\model\project\user.js

```
async function register (account, userInfo) {
  const tableName = getTableName();
}
```

### 7.4 页面性能 #

http://localhost:8080/project/1/monitor/performance (http://localhost:8080/project/1/monitor/performance)

#### 7.4.1 urlList #

- 用来提供某时间范围内的URL性能列表，因为在查看性能指标的时候会按页面的URL来进行，所以在页面的初始阶段会提供一个接口函数来获取 这些性能指标对应的URL列表

client\src\view\performance\index.vue

```
    async getUrlList (params = {}) {
      const {
        st,
        et,
        summaryBy
      } = params
      const res = await fetchUrlList({
        st: st || +this.dateRange[0],
        et: et || +this.dateRange[1],
        summaryBy: summaryBy || 'minute'
      })
      this.urlData = (res.data || []).map((item, index) => ({
        name: item
      }))
      this.url = _.get(this, ['urlData', 0, 'name'], '')
      this.getTimeDetail()
      this.getTimeLine()
      if (this.urlColumns) {
        this.urlLoading = false
      }
    },
```

client\src\api\performance\index.js

```
export const fetchUrlList = (params) => {
  return axios.request({
    url: `project/${getProjectId()}/api/performance/url_list`,
    method: 'get',
    params: {
      ...params
    }
  })
}
```

```
Request URL: http:
```

server\src\routes\api\performance\index.js

```
let urlList = RouterConfigBuilder.routerConfigBuilder('/api/performance/url_list', RouterConfigBuilder.METHOD_TYPE_GET, async (req, res) => {
  let projectId = _.get(req, ['fee', 'project', 'projectId'], 0);
  console.log('projectId ',projectId);
  let request = _.get(req, ['query'], {})

  let startAt = _.get(request, ['st'], 0)
  let endAt = _.get(request, ['et'], 0)
  const summaryBy = _.get(request, 'summaryBy', '')
  if (_.includes([DATE_FORMAT.UNIT.DAY, DATE_FORMAT.UNIT.HOUR, DATE_FORMAT.UNIT.MINUTE], summaryBy) === false) {
    res.send(API_RES.showError(`summaryBy参数不正确`))
    return
  }

  const currentStamp = moment().unix()

  if (startAt) {
    startAt = _.floor(startAt / 1000)
  } else {
    startAt = currentStamp
  }
  if (endAt) {
    endAt = _.ceil(endAt / 1000)
  } else {
    endAt = currentStamp
  }
  let urlList = await MPerformance.getDistinctUrlListInRange(projectId, MPerformance.INDICATOR_TYPE_LIST, startAt, endAt, summaryBy)
  res.send(API_RES.showResult(urlList))
}
)
```

server\src\model\parse\performance.js

```
async function getDistinctUrlListInRange (projectId, indicatorList, startAt, endAt, countType = DATE_FORMAT.UNIT.MINUTE) {
   for (let tableName of tableNameList) {
    console.log('tableName,countType,indicatorList,countAtTimeList',tableName,countType,indicatorList,countAtTimeList);
    let rawRecordList = await Knex
      .distinct(['url'])
      .from(tableName)
      .where({
        count_type: countType
      })
      .whereIn('indicator', indicatorList)
      .whereIn('count_at_time', countAtTimeList)
      .catch((e) => {
        Logger.warn('查询失败, 错误原因 =>', e)
        return []
      })
    for (let rawRecord of rawRecordList) {
      if (_.has(rawRecord, ['url'])) {
        let url = _.get(rawRecord, ['url'])
        urlList.push(url)
      }
    }
  }
}
```

#### 7.4.2 lineChartData #

- 在获取 URL之后，根据对应的时间参数提供参数时间范围内的提定URL下面的所有指标折线图

client\src\view\performance\index.vue

```
async getTimeDetail (params = {}) {
      const {
        st,
        et,
        url,
        summaryBy
      } = params
      const res = await fetchTimeDetail({
        st: st || +this.dateRange[0],
        et: et || +this.dateRange[1],
        url: url || this.url,
        summaryBy: summaryBy || 'hour'
      })
      const dv = new DataSet.View().source(res.data)
      dv.transform({
        type: 'rename',
        map: {
          dns_lookup_ms: 'DNS查询耗时',
          response_request_ms: '请求响应耗时',
          dom_parse_ms: 'DOM解析耗时',
          response_transfer_ms: '内容传输耗时',
          load_resource_ms: '资源加载耗时',
          dom_ready_ms: 'DOM_READY_耗时',
          first_render_ms: '首次渲染耗',
          first_response_ms: '首次可交互耗时',
          first_tcp_ms: '首包时间耗时',
          load_complete_ms: '页面完全加载耗时',
          ssl_connect_ms: 'SSL连接耗时',
          tcp_connect_ms: 'TCP链接耗时'
        }
      })
      dv.transform({
        type: 'fold',
        fields: [
          'DNS查询耗时',
          '请求响应耗时',
          'DOM解析耗时',
          '内容传输耗时',
          '资源加载耗时',
          'DOM_READY_耗时',
          '首次渲染耗',
          '首次可交互耗时',
          '首包时间耗时',
          '页面完全加载耗时',
          'SSL连接耗时',
          'TCP链接耗时'
        ],
        key: 'type',
        value: 'ms'
      })
      const data = dv.rows
      this.lineData = data
      const scale = [{
        dataKey: 'ms',
        sync: true,
        alias: 'ms',
        formatter: (value) => value + ' ms'
      }, {
        dataKey: 'index_timestamp_ms',
        type: 'time',
        tickCount: 10,
        mask: 'MM-DD HH:mm'
      }]
      this.lineScale = scale
      if (this.lineData && this.lineScale) {
        this.isSpinShowDetail = false
      }
    },
```

```
export const fetchTimeDetail = (params) => {
  return axios.request({
    url: `project/${getProjectId()}/api/performance/url/line_chart`,
    method: 'get',
    params: {
      ...params
    }
  })
}
```

```
http:
```

server\src\routes\api\performance\index.js

```
let lineChartData = RouterConfigBuilder.routerConfigBuilder('/api/performance/url/line_chart', RouterConfigBuilder.METHOD_TYPE_GET, async (req, res) => {
  let projectId = _.get(req, ['fee', 'project', 'projectId'], 0)
  let request = _.get(req, ['query'], {})
  res.send(API_RES.showResult(resultList))
}
)
```

server\src\model\parse\performance.js

```
async function getDistinctUrlListInRange (projectId, indicatorList, startAt, endAt, countType = DATE_FORMAT.UNIT.MINUTE) {
```

#### 7.4.3 urlOverview #

- 在获取 URL之后，可以根据此接口提供的时间范围指定的URL的各项指标平均值

client\src\view\performance\index.vue

```
async getTimeLine (params = {}) {
      const {st,et} = params
      const res = await fetchTimeLine({
        st: st || +this.dateRange[0],
        et: et || +this.dateRange[1],
        url: this.url,
        summaryBy: 'minute'
      })
```

client\src\api\performance\index.js

```
export const fetchTimeLine = (params) => {
  return axios.request({
    url: `project/${getProjectId()}/api/performance/url/overview`,
    method: 'get',
    params: {
      ...params
    }
  })
}
```

```
http:
```

server\src\routes\api\performance\index.js

```
let urlOverview = RouterConfigBuilder.routerConfigBuilder('/api/performance/url/overview', RouterConfigBuilder.METHOD_TYPE_GET, async (req, res) => {
  let projectId = _.get(req, ['fee', 'project', 'projectId'], 0)
  let request = _.get(req, ['query'], {});
}
```

```
async function getUrlOverviewInSameMonth (projectId, urlList, startAt, endAt, countType) {
}
```

- front-monitor (https://gitee.com/zhufengpeixun/front-monitor)
- node dist/fee.js Utils:TemplateSQL