

迭代二需求规格说明文档

Gitming Project



2016-2-28

NANJING UNIVERSITY

|  |  |  |  |
| --- | --- | --- | --- |
| 文档类别 | 需求规格说明文档 | 文档编号 | I2\_REQUIRMENTS\_1.0 |
| 创建人 | 崔浩、袁阳阳 | 创建时间 | 2016-02-28 |
| 最后一次修改人 | 袁阳阳 | 最后一次修改时间 | 2016-03-13 |

目录

[1. 引言 2](#_Toc445634970)

[1.1目的 2](#_Toc445634971)

[1.2范围 3](#_Toc445634972)

[1.3参考文献 3](#_Toc445634973)

[2. 总体描述 3](#_Toc445634974)

[2.1商品前景 3](#_Toc445634975)

[2.1.1背景与机遇 3](#_Toc445634976)

[2.1.2业务需求 4](#_Toc445634977)

[2.2商品功能 4](#_Toc445634978)

[2.3用户特征 4](#_Toc445634979)

[2.4约束 4](#_Toc445634980)

[2.5假设和依赖 4](#_Toc445634981)

[3. 详细需求描述 5](#_Toc445634982)

[3.1对外接口需求 5](#_Toc445634983)

[3.1.1用户界面 5](#_Toc445634984)

[3.1.2通信接口 5](#_Toc445634985)

[3.2功能需求 5](#_Toc445634986)

[3.2.1 GitHub用户信息查询 5](#_Toc445634987)

[3.2.2 GitHub仓库信息查询 7](#_Toc445634988)

[3.2.3 GitHub用户信息统计 8](#_Toc445634989)

[3.2.4 Github仓库信息统计 10](#_Toc445634990)

[3.3非功能需求 11](#_Toc445634991)

[3.3.1 可维护性 11](#_Toc445634992)

[3.3.2 可靠性 11](#_Toc445634993)

[3.4数据需求 11](#_Toc445634994)

[3.4.1 数据定义 11](#_Toc445634995)

[3.4.2 默认数据 12](#_Toc445634996)

## 1. 引言

### 1.1目的

本文档描述了Github数据分析系统Gitming的功能需求和非功能需求。开发小组的软件系统的实现和验证工作都以此文档为依据。

### 1.2范围

Github数据分析系统Gitming是为分析Github项目与用户信息开发的业务系统，开发的目标是帮助编程爱好者找到自己感兴趣的项目并了解该项目的一些信息，包括项目、用户信息的查询。

### 1.3参考文献

1）IEEE标准。

2）Github数据分析系统Gitming用例文档V1.0。

## 2. 总体描述

### 2.1商品前景

#### 2.1.1背景与机遇

GitHub系统是全球最大的社交编程及代码托管网站，作为开源代码库以及版本控制系统，拥有140多万开发者用户，且随着应用程序向云上的转移，GitHub已经成为了管理软件开发以及发现已有代码的首选方式。但是随着GitHub的普及，项目、仓库、用户的增多，在GitHub上找到编程爱好者感兴趣且适合的项目变得越发困难。希望能够更好更快地为开发者和编程爱好者找到合适的项目、GitHub用户，并查看项目和用户的相关信息。

GitHub数据分析系统Gitming就是为了满足GitHub网站这一需要而开发的，它通过客户端从网页获取API数据，并进行显示、分析，用户通过客户端查询需要的项目或GitHub用户信息。

#### 2.1.2业务需求

BR1：在系统开放1个月后，用户查找项目或GitHub用户的速度提升20%.

### 2.2商品功能

SF1：快速显示用户需要的项目与用户信息。

SF2：显示github用户和项目统计信息

### 2.3用户特征

本系统由开发者或编程爱好者使用，他们在GitHub网站上查找感兴趣或者自己擅长的项目以及GitHub用户的信息，由于GitHub内容的多样性或对自己的不够了解，他们难以找到适合的项目及信息，希望系统可以帮助他们解决这些问题。

### 2.4约束

CON1：项目整体采用Java语言开发。

CON2：系统使用的是PC端的图形界面。

CON3：迭代一迭代二不使用数据库。

CON4：工程行为记录在GitLab上。

### 2.5假设和依赖

<http://www.gitmining.net/api>不会崩溃（迭代二中在数据初始化后这一依赖已经无关紧要。）

## 3. 详细需求描述

### 3.1对外接口需求

#### 3.1.1用户界面

#### 3.1.2通信接口

### 3.2功能需求

#### 3.2.1 GitHub用户信息查询

3.2.1.1特性描述

在用户使用系统时，输入关键字或者根据标签查找GitHub用户信息，完成对符合要求的用户信息的显示。

3.2.1.2刺激/响应序列

刺激：用户选择查看GitHub用户信息

响应：系统显示单页GitHub用户信息列表

刺激：用户输入关键字并确认检索GitHub用户

响应：系统显示符合关键字要求的GitHub用户列表

刺激：用户选择具体的GitHub

响应：系统显示该用户具体信息。包括展现用户能力的雷达图

刺激：用户在github用户信息列表中选择同页内多位用户进行对比

响应：系统显示被选择用户各个参数信息对比的表格和对比的雷达图

3.2.1.3相关功能需求

|  |  |
| --- | --- |
| Gitusers.input  Gitusers.Input.Search | 系统要允许用户在查询时使用鼠标和键盘输入 |
| 在用户输入关键字并检索后，系统要显示符合关键字要求的GitHub用户列表，无符合要求项时显示为空 |
| Gitusers.Show | 用户选择查看GitHub用户时，系统显示所有GitHub用户信息列表 |
| Gitusers.Show.detail | 用户选择某一具体GitHub用户时，系统显示该GitHub用户具体信息，包括id，login，type，name，company，blog，location，email，bio， public\_repos，public\_gists，followers，following，created\_at，updated\_at |
| Gitusers.Show.detail.Project | 用户选择查看某一GitHub用户的相关项目，系统显示该GitHub用户相关联的项目列表，包括该用户拥有的、订阅的、合作的、贡献的项目。 |
| Gitusers.compare | 系统允许用户选择两个github用户进行对比 |
| Gitusers.compare.show | 用户选择两个github用户进行对比后，系统显示两位用户参数的对比表格和对比的能力雷达图 |

#### 3.2.2 GitHub仓库信息查询

3.2.2.1 特征描述

用户可以查看项目列表，也可以选择某个具体的Github仓库查看它的详细信息。

优先级=高

3.2.2.2 刺激/响应序列

刺激：用户选择查看仓库信息

响应：系统提供单页仓库列表

刺激：用户输入想查看的不包括用户名的仓库名

响应：系统显示符合搜索条件的仓库

刺激：用户选择具体的仓库

响应：系统显示该仓库的具体信息，包括仓库各个属性的雷达图

刺激：用户选择同一页码内的多个仓库进行对比

响应：系统显示被选中仓库参数属性对比的表格和对比的雷达图

3.2.2.3 相关功能需求

|  |  |
| --- | --- |
| Repository. input | 系统要允许用户在查询时使用键盘输入 |
| Repository. input. Invalid | 系统要能在用户输入不合法时拒绝用户请求并说明不合法原因 |
| Repository. search | 系统要支持用户按仓库名称检索 |
| Repository. show | 系统要能展示仓库的基本信息，包括：项目主页、项目描述、最后更新时间、stars、forks、contributors数量。 |
| Repository. show. basic | 系统要能展示单个项目的基本信息，包括：项目语言、所有者（id、类型）、项目主页、项目描述、创建时间、最后更新时间、stars、forks、contributors数量、项目大小，并提供查询其他详细信息的入口 |
| Repository. show. owner | 系统要能显示单个项目所有者的id、登陆名和所有者类型 |
| Repository. show. contributor | 系统要能显示单个项目的所有contributor信息 |
| Repository. show. collaborator | 系统要能显示单个项目的collaborator信息 |
| Repository. show. branch | 系统要能显示单个项目的所有版本信息 |
| Repository. show. fork | 系统要能显示单个项目的所有fork项目，并提供查看该fork项目的仓库信息的入口 |
| Repository. show. commit | 系统要能显示单个项目最近5条commit信息、包括commit编号，commit内容，删除代码行数，新增代码行数，总计影响行数、更改文件数 |
| Repository. show. issue | 系统要能显示单个项目的所有issue编号及其内容 |
| Repository.compare | 系统允许用户在仓库信息列表选择两个仓库进行对比 |
| Repository.compare.show | 用户选中两个仓库进行对比后，系统显示两个仓库参数属性对比的表格和对比的雷达图。 |

#### 3.2.3 GitHub用户信息统计

3.2.3.1 特征描述

用户使用系统时，可以查看github用户的相关统计信息

优先级 = 高

3.2.3.2 刺激/响应序列

刺激：用户选择查看github用户统计信息

响应：系统显示可供选择查看的用户信息统计项，具体项参见用例文档——用户信息统计

刺激：用户选择具体要查看的统计项

响应：系统显示该项统计信息

3.2.3.3相关功能需求

|  |  |
| --- | --- |
| UserStatistic.input | 系统允许用户在查看时使用鼠标点选 |
| UserStatistic.showKinds | 用户选择查看用户统计信息后，系统显示可供选择查看的统计项，具体项参见UserStatistic.showSpecific |
| UserStatistic.showSpecific | 用户选择具体查看的统计项后，系统显示相应具体项的用户统计信息 |
| UserStatistic.showSpecific.orgUser | 系统显示真实用户和组织的统计 |
| UserStatistic.showSpecific.comUserNum | 系统显示部分公司的github用户数量统计 |
| UserStatistic.showSpecific.blog | 系统显示用户所用博客的统计 |
| UserStatistic.showSpecific.location | 系统显示用户所在地统计 |
| UserStatistic.showSpecific.email | 系统显示用户使用的电子邮件统计 |
| UserStatistic.showSpecific.createTime | 系统显示用户创建时间统计 |
| UserStatistic.showSpecific.repoNum | 系统显示用户仓库数量统计 |
| UserStatistic.showSpecific.gistNum | 系统显示用户gist数量统计 |
| UserStatistic.showSpecific.followersNum | 系统显示用户follower数量统计 |
| UserStatistic.showSpecific.followingNum | 系统显示用户following数量统计 |
| UserStatistic.showSpecific.orgUserRepo | 系统显示组织的用户和仓库统计 |

#### 3.2.4 Github仓库信息统计

3.2.4.1 特征描述

用户使用系统时，可以查看github仓库的相关统计信息

3.2.4.2 刺激/响应序列

刺激：用户选择查看github仓库统计信息

响应：系统显示可供选择查看的仓库信息统计项，具体项参见用例文档——仓库信息统计

刺激：用户选择具体要查看的统计项

响应：系统显示该项统计信息

3.2.4.3 相关功能需求

|  |  |
| --- | --- |
| RepoStatistic.input | 系统允许用户使用鼠标点选 |
| RepoStatistic.showKinds | 用户选择查看仓库统计信息后，系统显示可供选择查看的统计项，具体项参见RepoStatistic.showSpecific |
| RepoStatistic.showSpecific | 用户选择具体查看的统计项后，系统显示相应具体项的仓库统计信息 |
| RepoStatistic.showSpecific.language | 系统显示项目使用的语言统计 |
| RepoStatistic.showSpecific.createTime | 系统显示项目创建时间的分布 |
| RepoStatistic.showSpecific.contributorNum | 系统显示项目的贡献者人数统计 |
| RepoStatistic.showSpecific.collaboratorNum | 系统显示项目的合作者人数统计 |

### 3.3非功能需求

#### 3.3.1 可维护性

1. 如果系统的交互语言（中文、英语）要发生变化，系统要在0.1人月内完成修改。
2. 如果系统获取的数据格式发生变化，系统要在0.5人月内完成修改。

#### 3.3.2 可靠性

1、如果网络连接中断，系统不能崩溃。

2、如果网络连接中断，系统要给出提示信息。

### 3.4数据需求

#### 3.4.1 数据定义

**DR1：GIT用户（User）（owner、contributor、collaborator）**

包括：id，login，type，name，company，blog，location，email，bio， followers，following，创建时间，最后更新时间，以及相关联的[项目/仓库(repository)](#_DR2：项目/仓库（repository）)

**DR2：项目/仓库（repository）**

包括：项目语言（language）、[所有者(owner)](#_DR3：所有者（owner）)、项目主页（url）、项目描述、创建时间、最后更新时间、stars、forks、contributors数量、项目大小

以及：[contributor](#_DR1:GIT用户（User）)信息、[collaborator](#_DR1:GIT用户（User）)信息、[版本](#_DR4：版本（branch）)信息、fork[项目](#_DR2：项目/仓库（repository）)信息、[commit](#_DR4：提交（commit）)信息、[事件（issue）](#_DR5：事件（issue）)信息

**DR3：所有者（owner）**

所有者有该[用户](#_DR1:GIT用户（User）（owner、contributor、c)的信息，以及所有者类型（owner\_type：（分为User和Organization））.这是user的一个枚举属性。

**DR4：提交（commit）**

包括日期与所属仓库，以键值对的方式存储。（repositoryname->List<created\_at>）

**DR5：事件（issue）**

包括所属仓库与创建时间（created\_at），以键值对的方式存储。（repositoryname->List<created\_at>）

#### 3.4.2 默认数据

1、初始项目列表按被关注（star）度排序。

2、初始用户列表按被关注（followed）度排序。