文档编号：“面筋”APP– DSN – 1.0.0

“面筋”面试经验分享 APP
软件设计规格说明书

日期：2015-07-15

文档变更历史记录

序号	变更日期	变更人员	变更内容详情描述	版本
1	2015/07/15	潘浩彬	创建	0.9.9
2	2015/07/15	潘浩彬	增加部署图	1.0.0
3	2015/07/15	张秋怡	修正错误	1.1.0

目录

1、引言

1.1 编写目的

根据《需求规约说明书》等相关文档，综合考虑在实际生活场景的各种应用，本需求的编写目的在于平等地进行面试经验分享，倡导共同学习，共同成长的学习方式，使得求职者能够更好地去了解公司，找清楚自身的定位。

1.2 读者对象

本需求的预期读者是与本软件开发有联系的决策人，开发组成人员，扶助开发者，支持本项目的领导和公司，软件测试员。

1.3 软件项目概述

- 项目名称

  “面筋”面试经验分享平台

- 简称

  “MianJin”

- 项目代码托管地址

  https://github.com/joyeecheung/get-offer

- 软件项目的大致功能和性能要求

  本项目主要功能是以匿名论坛的方式来分享用户的面试经历，入职信息，实习/就业感受以及

知识交流。

在性能上并没有很高的要求，用户只需要通过访问该网站，并能轻松查询相关的帖子。

1.4 文档概述

该文档主要从软件分析和设计的方向来阐述整个软件开发的流程和思路，文档中包含了软件相关业务流程的介绍以及图示，还包含了软件的组织形式和相关配置。本次开发采用了敏捷开发的方式，通过不断迭代的方式，逐渐明确需求，着重强调代码的高耦合低内聚，以较好的组织方式对代码进行组织和重构。

1.5 定义

文档中采用的缩略词以及相关术语:

MJ : 该软件”面筋”，

Topic : 用户发布的话题，

Reply : 用户回复的消息，

Node : App 拥有的模块

1.6 参考资料

Nodejs	http://www.nodejs.org
Polymer	https://www.polymer-project.org/

Mongodb http://www.mongodb.org

2. 软件设计约束

2.1 设计目标和原则

本设计欲实现主要的用例有:

- 1. 用户通过加入不同板块来发表自己的经验和看法
- 2. 用户检索并查看相关板块，来获取信息
- 3. 用户可以发起回答或者发表留言表示感谢或质询

2.2 设计约束

硬件平台 : 客户端包括手机、电脑、平板等可运行提供支持的浏览器的硬件，服务器端为普通 PC 或 Mac。

OS 要求 : 客户端包括 iOS、Android、Windows、Mac OS X、Linux, 服务器可为支持 Node.js 与 MongoDB 的 Windows、Mac OS X 或 Linux 版本。

开发语言 : JavaScript、HTML、CSS

开发工具 : Gulp、Mocha、Supervisor、JSHint、chai 等

灵活性和配置要求:

服务器需要配置 Node.js 与 MongoDB 环境，支持任意支持这两个环境的操作系统。

客户端不限设备大小（可兼容手机、平板、电脑），只需

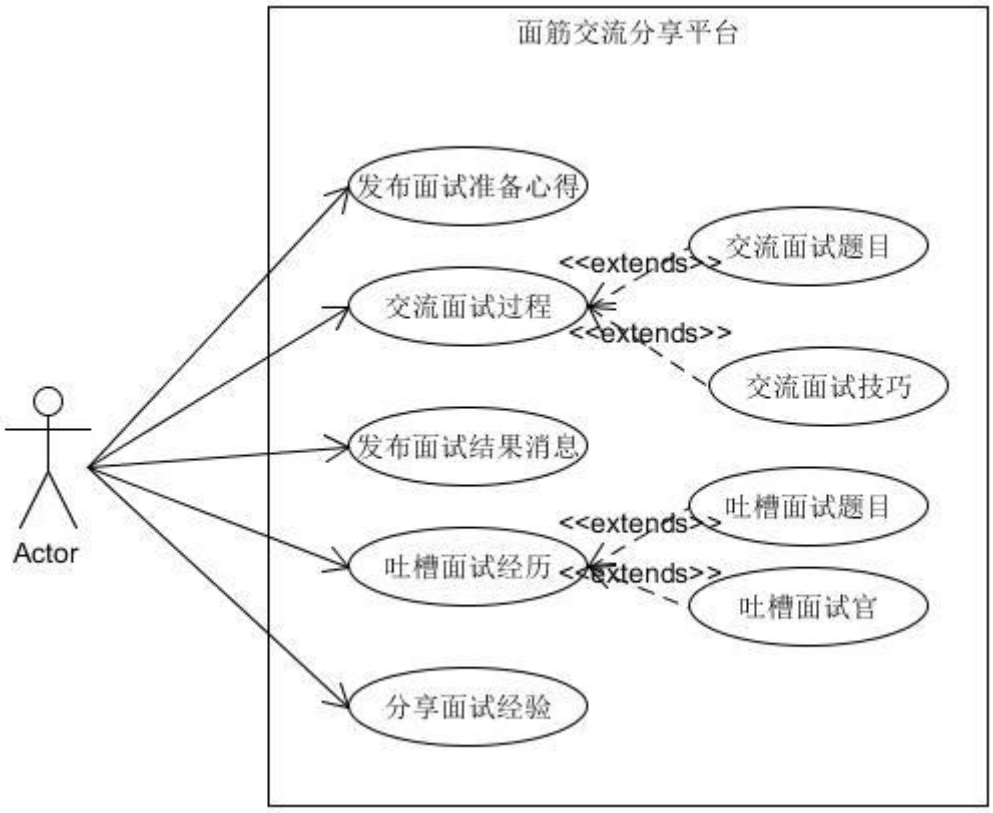
要支持包括移动端浏览器 （IE mobile 10+，iOS Safari 7+，Android 4.4+）及桌面端浏览器（

IE10+，Firefox 30+，Chrome 34+，Opera 23+，Safari 7+)中的任意一个即可。

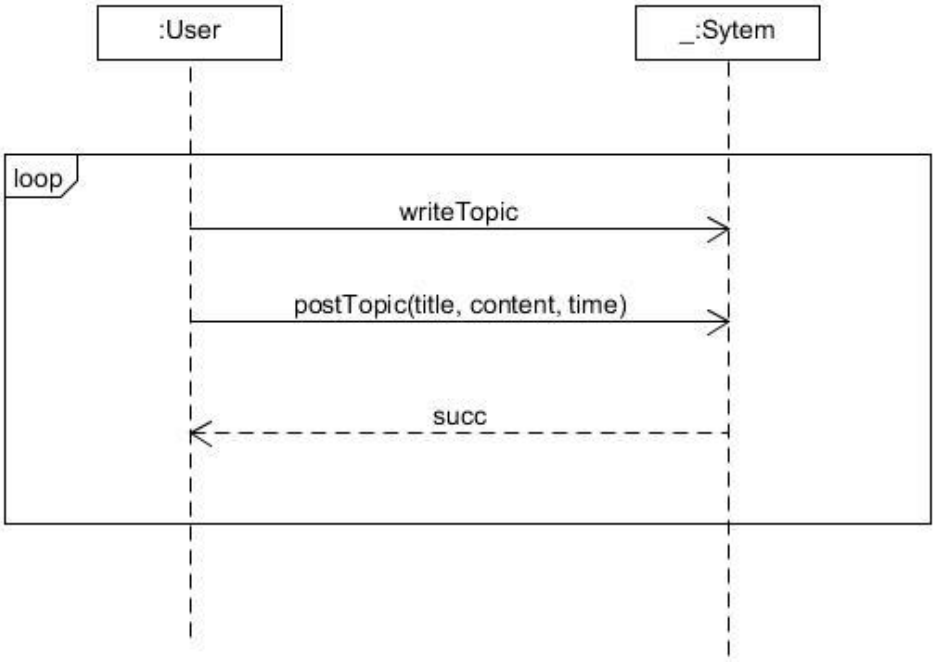
3.软件设计描述

3.1 总体结构设计

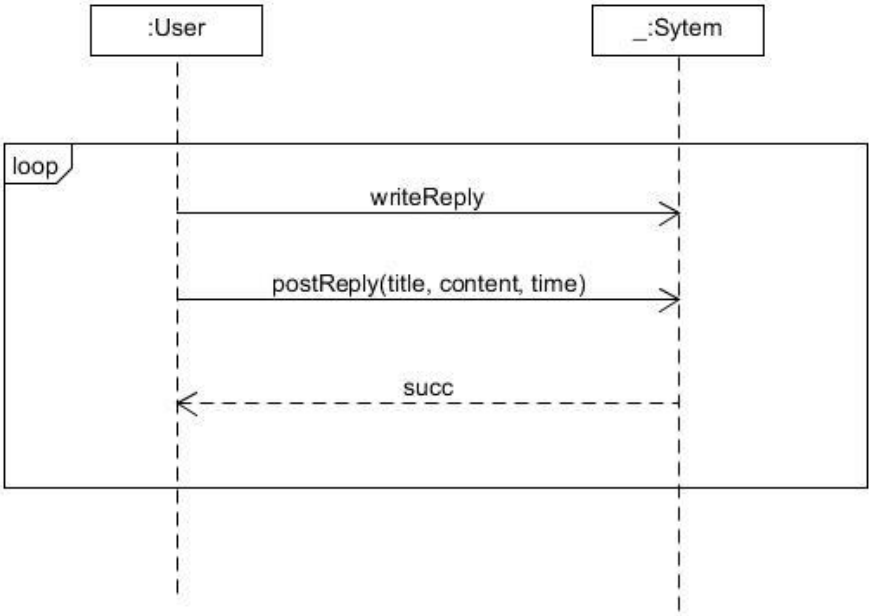
1. 根据用户的需求，绘制出系统用例



2. 根据用户的需求，绘制出系统顺序图

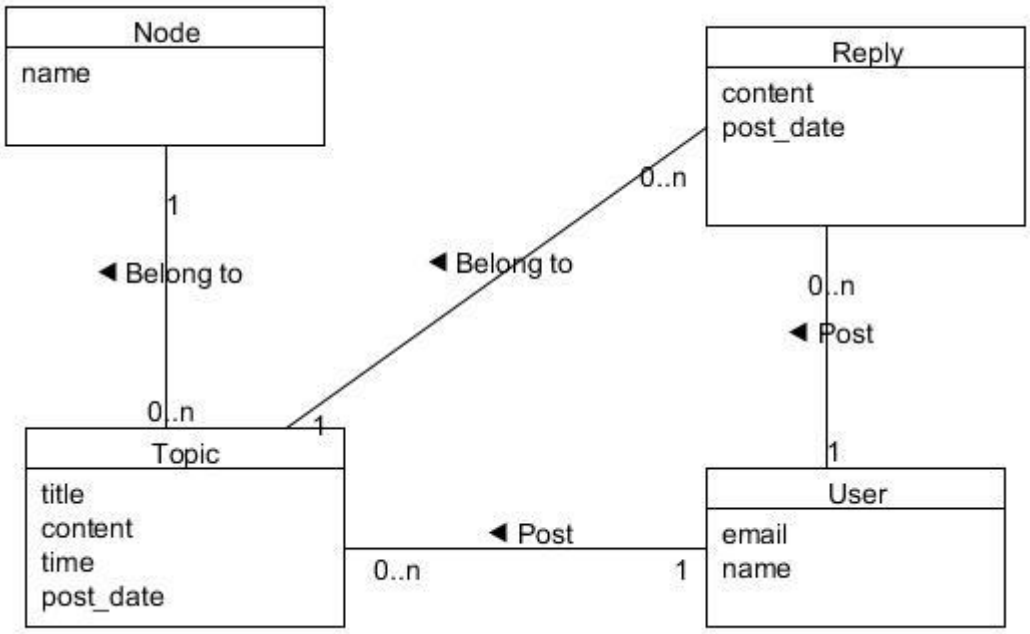


发布 Topic



用户回复

3.软件设计的领域模型:



3.2 数据设计

- 采用 MongoDB 为本次软件的数据存储，其弱一致性，能够保证用户的访问速度
- 文档结构的存储方式，能够更便捷的获取数据。
- 由 3.1 的领域模型，大致建立了如下 3 个数据库表

(1) node

字段:[

String name

]

(2) topic

字段:[

String title,

String content,

```
        String user_name,
        String user_email,
        String node_name,  ; 外键关联
        Date post_date,
        Date last_update,
        Int reply_count
        ]
(3) reply
        字段: [
        String content,
        String user_name,
        String user_email,
        ObjectId topic_id, ;  外键关联
        Date post_date
        ]
```

3.3 详细设计

设计主要分为前端，后台以及数据库、测试和开发环境。

前端设计:

前端采用 Polymer 与 Page.js 编写为响应式的 SPA。Polymer 是 Web Components 的预先实现，标志着 HTML 的未来，其组件化、模块化的开发方式有助于小组成员分工合作，并且为现代浏览器提供了一个精简的兼容性层，可帮助减少浏览器兼容性的工作量。Page.js 简单易用，方便小组成员快速上手。UI 库使用符合 Material Design 的系列组件（以 Paper Elements 为主），方便完成设计需求。与后端的数据交互基于<iron-ajax> 声明式 AJAX 元素完成。

后台设计:

后台在 Node.js 的平台上使用 Express 做开发，采用 MVC 架构以及 RESTful 架构风格，主要分为提供数据 API 及传输 SPA 所需静态文件两块。后台还编写了 markdown 以及 sanitize 中间件，过滤可能带有 XSS 攻击的数据，使得网站更为安全可靠。

数据库设计:

MJ 的数据以文档为主，并且存在非结构化的数据，适合使用非结构化的文档型数据库，并且对事务要求不高，因此采用了符合描述的数据库中最为成熟的 MongoDB，同时在 Node.js 中使用 Mongoose 作为 ODM，简化数据库操作。

测试设计:

为了测试接口的可用性，采用了 Mocha 与 Chai 来进行测试，测试了包括静态页面的渲染，Node API，Topic API 及 Reply API。

开发环境:

使用基于流的 Gulp 及系列组件辅助前端工作流，利用 Supervisor 辅助后端开发，不限制成员使用的 OS、编辑器或浏览器。

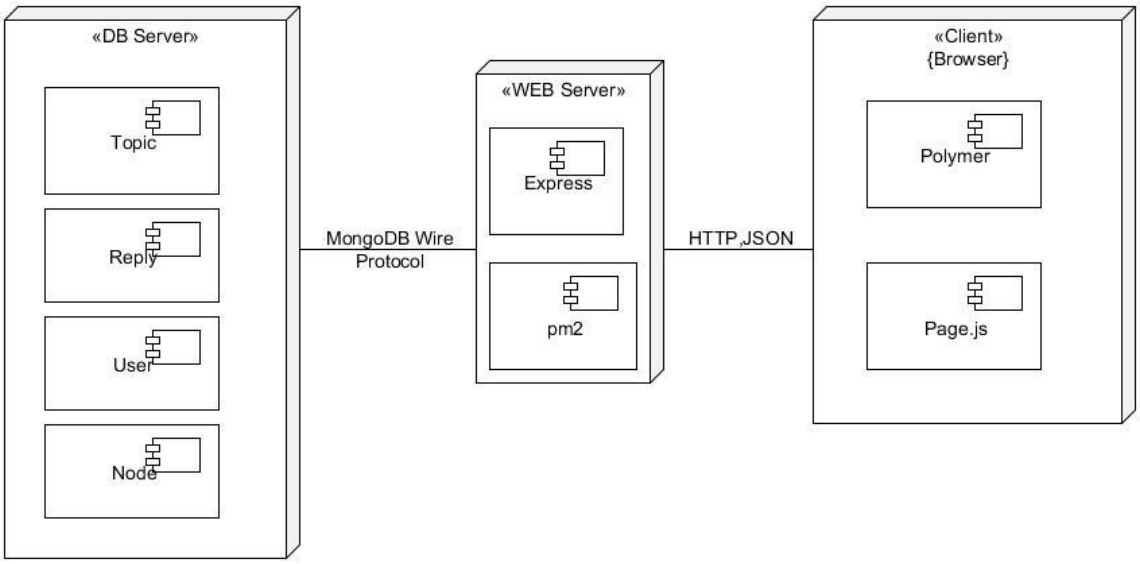
3.4 接口设计

- 在 MJ 设计的时候，考虑到兼容多平台的问题，因此在设计的时候，采用了 RESTful 风格，将前端与后台的通讯简化为 JSON 数据的传输，大大使得开发得以并行。

“面筋”的 API 大致如下:

GET	api/topic/:topic_id	获取该 topic_id 的 reply 等相关讯息
POST	api/topic json_data	发布 topic
GET	api/node/:name	获取板块 name 的最新 topic 等相关讯息
GET	api/reply/:reply_id	获取该 replay_id 对应的 reply 相关信息
POST	api/reply json_data	发布 reply

3.5 实际部署图


«DB Server»
Topic
Reply
User
Node
MongoDB Wire
Protocol
«WEB Server»
Express
pm2
HTTP,JSON
«Client»
{Browser}
Polymer
Page.js