南京大学

本 科 毕 业 设 计

院　系 软件学院

专　业 软件工程

题　目 kikbug 系统前端的设计与实现

年　　级 **121250008**

学生姓名 陈丹丹

指导教师 陈振宇　职　称 副教授

论文提交日期

南京大学本科生毕业论文（设计）中文摘要

毕业论文题目： kikbug 系统前端的设计与实现

软件学院 院系 软件工程 专业 2012 级本科生姓名：陈丹丹

指导教师（姓名、职称）：

摘要：

随着智能手机的普及，手机应用对于人们的日常生活来说变得越来越重要。手机应用的版本迭代更新变得越来越迅速，使用的场景也越发复杂多变。应用开发者往往来不及也没有相应的精力对移动应用做详尽全面的测试。互联网的发展促进了众包概念的流行，众包测试在最近几年也变得炙手可热。但是由于用户时间不稳定、水平参差不齐，众包测试有时候并不能保证提供有效的测试。计算机相关专业的高校学生专业水平较高且拥有较多的课外时间，同时也希望接触到产业界的实际工作来实践提升自己的专业技能，因此面向学生的众包测试平台是一个非常有前景的项目。

Kikbug 系统是一个面向大学教学的移动应用众包测试平台。开发者可以注册 kikbug 企业账号获得免费的众包测试服务，服务内容包括手动测试、自动测试和安全测试。kikbug 平台的运营人员将分解测试需求并推送给合适的测试员。Kikbug 的测试人员主要是来自慕测系统的学生，教师可以利用这些测试任务来作为课堂上的练习和课后的考试，当然测试人员的注册入口也向互联网开放。Kikbug 系统目前提供 Web、Android 和 IOS 等多种客户端版本。

Kikbug 因为要兼容移动端，所以采用前后端分离的方式进行开发。后端 api 使用 java 语言在 spring-hibernate 应用框架下进行开发，为 web 端和移动端提供统一的 restful 接口，Web 前端使用 angular.js 作为开发框架。本文将详细介绍 Kikbug 系统 Web 前端的设计与实现。

关键词：Kikbug；众测；移动应用；前端；AngularJS;

南京大学本科生毕业论文（设计）英文摘要

With the popularity of mobile devices, mobile apps becomes more and more important in our daily life. Mobile apps refreshes more quickly and they are used in more complicated environments. Developers often can't apply complete tests to their apps. The development of the Internet makes crowdsource becomes popular, crowdsourced testing also becomes popular in recent years. But because of lots of shortcomings of crowdsourced testing, sometimes it is difficult to ensure that all the app are well-tested. The technical level of university students is much higher than testers of crowdsourced testing platform users. So it's a nice try to make students in colleges engage crowdsourced testing.

Kikbug is a project for college education. Developers can register an Kikbug business account for free crowdsource testing service which include manual test、auto test and security test. Operators of kikbug will split app to be tested into several tasks and distribute them to suitable testers. Students from Mooc System is a major testers in kikbug, teachers can use kikbug's tasks as practices and exams in class. As well, The registration of testers is also open to the Internet. Currently web and android as well as ios clients are available.

To be compatible with the mobile terminal, we decompose the system into front end and back end.The back end is developed with the help of spring-hibernate frame, provides restful interfaces to web application and mobile clients. The web application is based on AngularJS.This paper mainly introduces the design and implementation of kikbug's web application.

KEY WORDS: Kikbug; Crowdsourced testing; Mobile Application;

Frontend; AngularJS

目 录

图目录

表目录

第一章 引言

1.1 项目背景

随着智能手机的飞速发展和不断普及，移动应用在人们生活中出现的频次越来越高，不少传统的网站服务提供者都认识到这一点并且开始从电脑端迁移到移动端，移动应用正逐渐在现代人的生活中扮演一个不可或缺的角色。互联网用户的激增使得移动应用的用户数达到一个前所未有的数量，并且由于移动应用的使用场景比传统网页应用更加的复杂,同一个移动应用的使用环境包含了不同的操作系统版本、不同的硬件设备、不同的网络环境，在这样复杂的环境中确保应用一直运行正确是一件非常有挑战性的任务。对于个人开发者和一部分企业来说，购买种类齐全的测试设备和建立专业的测试团队是件不太可能或者成本过高的事情。面对这些难题，通过大量真实用户和行业专家来测试 app 在现实条件下的种种表现，这种被称为“众包测试”的新型测试服务正成为当下新蓝海。

1.2 国内（外）移动应用众包测试现状

众包测试目前在国外颇为流行，虽然这种理念传入国内年代不久，但由于需求量的巨大以及良好的发展前景，目前已有不少公司和开发团队在这一领域试水，目前已有不少颇具规模的众测服务提供商了，例如：testin、百度开放云移动 App 测试、腾讯众测平台等。这里主要分析一下 testin 和百度开放云移动 App 测试平台。

Testin 平台根据不同用户的需求为其提供多种的产品服务，包括内测、功能测试、兼容测试、崩溃分析、远程真机调试等。并且根据行业类型为用户提供整体的解决方案，目前已能为金融行业和手游行业提供整体的解决方案。其测试用例都是经过专家详细沟通后设计的，覆盖性全面，并且由专业的测试团队跟进，最终为用户提供详尽的测试报告。

百度开放云移动 App 测试平台为用户提供了多种测试选择，包括人工测试、自动化测试和用户评测。与 testin 不同的地方就在于，百度开放云不仅提供应用本身功能、安全等方面的测试，更近一步的提供用户评测，关注用户体验，用户定位和用户需求评测，可以帮助移动应用更好的赢取用户。

1.3 项目背景研究总结

这些众测平台目前已经吸引了不少用户，但其中大部分提供的都是有偿服务，并且除了平台专门组织的测试小组以外其他人数众多的测试员并不能保证其专业性。

Kikbug 众测平台与上述平台的最大区别就是 kikbug 的最主要测试人员是广大高校的计算机专业相关学生，这保证了测试结果的专业性和高质量。并且 Kikbug 通过引入众包测试的概念和实践进入大学教学当中，可以为教师和学生带来更丰富的教学体验。当然 kikbug 的测试人员入口也向互联网开放，提供群组功能，欢迎互联网测试人员及团队参与。

Kikbug 另一个优点就是我们的众测服务对于开发者而言是免费的，开发者只需要提供待测应用已经需求并且选择测试内容（手动测试，自动测试，安全测试），平台运营人员会根据这些信息安排具体的测试任务和计划，推送到合适的群组，安排测试并且最后除了详细测试报告外还会提供总结报告。

1.4 论文的主要工作和组织结构

kikbug 系统包括了服务器端，网页端和移动端，本文将主要介绍网页端的设计与实现，详细描述前端项目在 angular 框架下的组织结构与具体实现。

第一章：引言部分，包括了项目背景分析，同类型项目分析以及项目总结内容等。

第二章：介绍了 kikbug 前端项目中主要使用的框架和技术概览

第三章：介绍了 kikbug 的需求分析和概要设计。

第四章：介绍了 kikbug 前端项目的代码组织结构以及在 angular 框架下的前端模块组织。

第五章：详细描述了前端项目的实现细节。

第六章：项目成果总结，探讨未来的发展前景和改进。

第二章 技术概述

2.1 ANGULARJS

AngularJS 是一个专为创建 web 应用设计的开源 web 应用框架。 Angular 使用纯客户端 JavaScript 语言为开发者提供了 MVC、MVVM、 MVW、数据绑定、 依赖注入和出色的场景测试等功能。能够使客户端与服务端解耦，实现了并发处理机制并增强了代码的复用性。

AngularJS 基于声明式编程模式，用户可以基于业务逻辑进行开发。该框架基于传统的 HTML 并扩展了其内容，提供双向数据绑定，从而完成了数据自动同步机制以避免直接 DOM 操作带来的代码难以追踪调试问题。

对于创建一个富客户端应用来说，AngularJS 是一个组织良好，经过严格测试，功能丰富、强大而灵活的 javasrcipt MVC 框架。

Kikbug 的前端项目使用 angular 来构建应用，充分利用了其提供的指令、服务、控制器、路由、数据绑定等功能。

2.2 BOOTSTRAP

Bootstrap 是推特推出的当前最受开发者欢迎的前端框架。Bootstrap 开放源码,支持 CSS3 制定的全部属性和标准,支持 HTML5 的标签和语法，提供定制的 jQuery 插件并且已经能够支持所有的主流浏览器。 Bootstrap 的目的是提供一个直观、简洁、移动设备优先的响应式前端开发框架,让 web 开发更简单、迅速。Bootstrap 的某些样式使用 LESS 写就,可以让用户自己设置变量值以实现定制化。在样式之外,Bootstrap 还提供了十几个常用的 JavaScript 插件,如模态对话框、下拉菜单、滚动监听、标签页、 工具提示等 jQuery 插件。

Kikbug 的前端项目使用 bootstrap 作为整体风格框架，提供友好的页面风格和响应式的布局。

2.3 SASS

SASS 是一种动态样式语言，它扩展了 CSS3 为 CSS 赋予了动态语言的特性，诸如变量、运算、继承和函数等，它也是目前最成熟稳定、功能强大的 css 扩展语言。Sass 提供了许多种便利样式的写法，很大程度上节省了设计者的开发时间，使得 CSS 的开发和维护，变得更加简单。

Kikbug 系统的 css 文件全部使用 sass 来编写，并且使用 node 提供的 sass 编译工具来进行编译。

2.4 GULP

gulp.js 是一个基于流前端自动化构建工具。它有一下几个优点，首先它非常的易于使用，通过代码优于配置的策略，Gulp 使得简单的任务简单，复杂的任务可管理。其次是构建快速，它利用 Node.js 流的威力，可以使你快速构建项目并且减少频繁的 IO 操作。并且 Gulp 提供的插件质量都非常高，能够确保如你期望的那样简洁高质量完成任务。

Kikbug 前端项目使用 gulp 作为构建工具，结合 sass，browserify 等 css、js 编译工具，构建出生产环境下使用的前端代码。

2.5 BROWSERIFY

Browserify 能够让你使用类似于 node.js 的 require() 方式来组织浏览器端的 JS 代码，通过预编译让前端 JS 代码可以直接使用 Node 的 NPM 安装的一些库。同一份代码可以同时运行在服务器端和浏览器端，提高代码的复用性。并且这种方法使得前端 JavaScript 代码组织结构更加清晰，易于开发和维护。

Kikbug 前端项目使用 browserify 来组织 JavaScript 的编写格式和引用方式，使得最后的 JavaScript 文件都编译到一个单一的 dist.js 文件中，减少浏览器加载大量 JavaScript 文件的时间。

2.6 本章小结

在开发 Kikbug 前端应用的时候采用了前后端分离的方式，这是之前没有接触过的模式，开发过程中学习到了很多前端的知识、工具和框架，这一章就主要介绍了这些在 Kikbug 前端应用开发中使用到的相关技术。

第三章 系统需求分析和概要设计

3.1 项目整体概述

Kikbug 系统是一个面向教育的移动应用众包测试平台。它的主要目标是对接企业和高校的资源，帮助企业获得免费专业的测试服务以及帮助高校计算机专业的学生获得课内学习技能的实践以及提升。

Kikbug 系统对接了慕测系统，无缝接入了慕测平台的学生、教师、班级以及考试等概念，慕测用户无需重复操作即可立即使用 Kikbug 系统。

为了满足测试的充分性，Kikbug 除了发布手动测试任务以外，还接入了自动测试平台和安全测试平台，企业用户发布待测应用时只要进行对应项的选择，就可获得对应的服务，并且可以非常方便的查看测试结果。

当然，只以学生作为测试的执行者是存在运营时间上的不连续性的，所以 Kikbug 平台的测试人员注册入口也对来自互联网的众包测试员进行开放。并且新一版的 kikbug 新增了测试群组的功能，可以让一群有相同目标和交集的测试员聚集在一个群组中，小范围内集中的进行应用的测试，这样可以使得测试需求的执行更加集中和快速。

3.2 需求分析

Kikbug 的测试流程主要是这样一个过程：系统接受企业提供的待测应用，由管理员进行审核和推送，群主进行需求分解，测试员（主要为高校学生）进行测试并上传测试报告，测试结束后由群主进行测试结果总结。最终的测试结果包括细分报告、总结报告、测试人数、机型分布、安全和自动测结果将显示给企业用户。当然因为 kikbug 大部分的测试员都是高校计算机学生，提供的测试服务都是免费的，为了双向互利，kikbug 将对测试员的每份测试报告进行打分，这里的分数来源有三处，分别为来自群主的打分、提供应用的企业用户的打分和评价以及 kikbug 系统的自动打分。

3.2.1 用例描述

从上述的流程中可以看出，Kikbug 系统的涉众包括四类，分别是企业用户、测试用户、群主和管理员，他们的职责和任务如图 3.1 所示，图 3.1 是 kikbug 系统涉众的用例分析图。

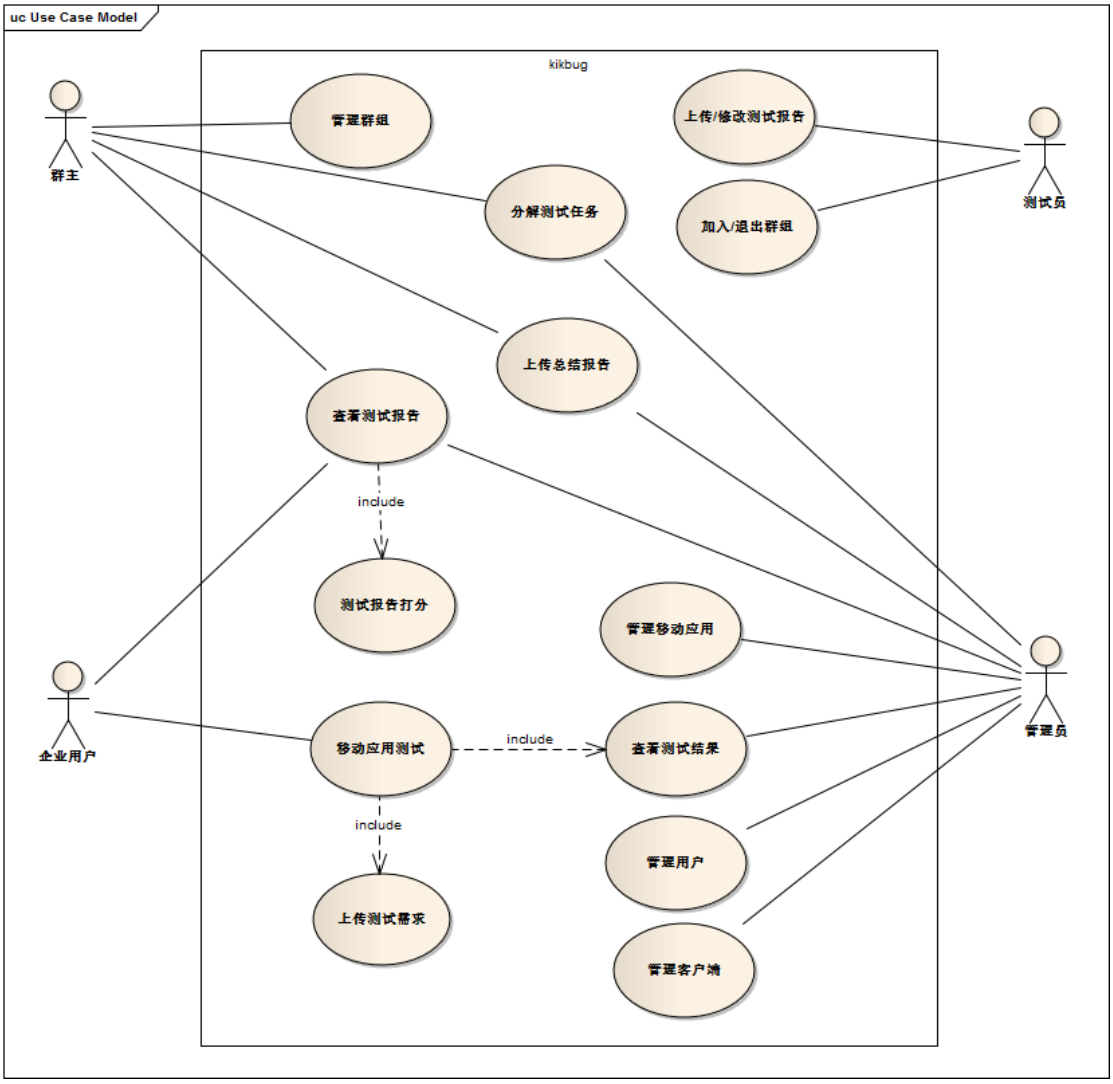

图 3.1 kikbug 系统用例积极图

1、企业用户

企业用户注册后通过平台验证后即可使用平台的众包测试服务。企业用户可以上传待测应用，并且选择测试方式，目前提供手动测试、安全测试和自动测试。测试过程中以及结束后，企业用户都可以查看测试的进程、报表、报告和总结。企业用户还可以为测试员上传的测试报告进行打分和评价。表 3.1 为企业用户的需求描述。

表 3.1 企业用户需求描述

需求名称	需求描述	优先级
创建应用	企业用户创建待测应用，包括基本信息、应用文件、测试需求文档等	高
查看应用列表	企业用户查看自己上传的全部待测应用	高
修改待测应用	在待测应用未审核或者审核不通过时，企业用	高

	户可以修改待测应用的内容	
查看测试结果	企业用户可以看见自己上传的应用的测试结果，包括统计报表、总结报告和细分报告。	高
给测试报告反馈	企业用户应当为测试员提交的测试报告评分和写评价	低

2、测试用户

Kikbug 平台的测试用户可以分为两类，一类是对接慕测平台产生的学生测试者，另一类是互联网测试人员。测试人员可以使用 Kikbug 提供的网页端和移动客户端查看推送的任务并接受，在移动客户端上进行测试，客户端会自动收集测试信息并上传生成测试报告，测试员还可以稍后再网页端进行报告的修改和完善。Kikbug 平台会根据测试报告，提供企业用户打分和评价、群主打分、自动打分等反馈信息，帮助测试用户了解自己的测试结果的质量、提升自我测试技能。测试用户的需求列表如表 3.2 所示。

表 3.2 测试用户需求描述

需求名称	需求描述	优先级
查看任务广场	测试员可以查看任务广场里的任务列表，并查看任务详情	高
接受任务	测试员可以选择自己喜欢的测试任务并且接受，然后就可以进行任务的测试了	高
上传报告	用户进行测试的过程中客户端会引导用户填写报告内容，测试结束后进行上传	高
修改完善报告	用户可以在网页上查看自己的报告并且修改和完善	中
加入群组	用户通过输入群号和群密码能够加入群组	中
查看群任务	用户可以查看自己加入的群里的任务	中
退出群组	用户可以自由的退出群组	低

3、测试群主

Kikbug 提供群组的概念，用户可以申请注册群主账号，通过审核后可以创建测试组，吸纳符合要求的测试员进入，更好的、更集中的完成测试任务。管理员会结合线下沟通的方式，将特定的测试应用推送到群组，使该测试需求只在群组内可见。Kikbug 的群组概念对应慕测系统的班级概念，群主对应慕测系统的

教师，因此慕测的教师用户可以无缝的使用 Kikbug 系统发布测试任务用做学生的考试、练习。群主用户的需求列表如表 3.3 所示。

表 3.3 群主用户需求描述

需求名称	需求描述	优先级
创建群组	群主可以创建一个群组，提供群口令	中
移除组内成员	群主有权限移除组内的某个成员	低
添加成员	群主有权限添加一个测试员到小组内	低
查看推送列表	群主可以查看管理员推送到群组中的待测应用	中
分解测试任务	群主需要对推送到群组中的应用进行任务分解	中
提交总结报告	群主应该要在某个应用测试结束后对其进行总结并上传总结报告	中
测试报告反馈	群主要对测试员提交的测试报告进行评分	低

4、管理员

管理员负责平台的日常运营管理事务，包括用户的审核管理、群组的审核管理、待测应用的审核管理、客户端的上传和更新等工作。

管理员的账号不可自行注册、由平台进行统一管理。管理员的需求列表如表 3.4 所示

表 3.4 管理员需求描述

需求名称	需求描述	优先级
审核群组	对于群主创建的群组，管理员需要进行审核	中
管理用户	对于新注册的企业用户和群主用户，管理员需要进行资质审核	高
审核应用	管理需要对企业用户提交的待测应用进行内容审核	高
推送待测应用	管理员要将通过审核的待测应用推送到任务广场或者合适的群组中，二者二选一不可并存，测试小组可多个	高
分解测试任务	管理员需要对推送到任务广场中的应用进行任务分解	高
提交总结报告	管理员应该要在某个推送到任务广场的应用测试结束后对其进行总结并上传总结报告	中
查看测试结果	管理员可以查看平台上的应用的测试结果，包括统计报表、总结报告和细分报告。	高

客户端管理	客户端的上传和更新管理需要管理员来维护	中

3.2.2 功能需求

Kikbug 系统的需求是基于第一版的运营情况进行的修改和完善的，第二版的需求一开始就已经较为明确。由于第二版的需求和第一版的差异较大，所以为了系统开发的清晰、后续维护的简便，Kikbug 的第二版项目完全抛弃了第一版的代码，从零开始开发。本节主要介绍 Kikbug 系统第二版的主要功能需求的详细流程。

1、注册

企业用户、测试用户和群主用户都需要注册后方可使用，其中企业用户和群组用户注册成功后还需要通过管理员的审核方可正式使用。为了方便与慕测对接，由慕测提供统一的账户管理系统，慕测系统中的学生和教师用户对于 Kikbug 中测试和群主账号，这些账户信息都在慕测端进行统一管理。

参与角色	企业用户、测试用户、群主用户
优先级	高
前置条件	注册手机号或者邮箱有效且尚未注册过
交互流程	1. 用户发起注册 2. 系统验证手机号或者邮箱合法，并且发送验证短信或邮件 3. 用户输入验证信息以及其他注册信息 4. 系统执行注册操作，并返回注册结果信息
后置条件	系统存在该用户注册信息

2、登陆

测试用户、超级管理员、审核通过的企业用户和群主用户以及慕测系统的学生和教师账号可以在 kikbug 系统直接登录。

参与角色	企业用户、测试用户、群主用户、管理员
优先级	高
前置条件	账号注册成功并且通过验证
交互流程	1. 用户填写登录信息 2. 系统执行登录操作，如成功则为其生成 session 信息并与用户 ID 和角色存入缓存中并返回给客户端。失败则告知原因。 3. 登录成功的话，客户端将服务器端返回的 session 信息本地缓

	存。
后置条件	登录成功系统则存储该用户的 session 信息，有效时间为 24 小时

3、企业用户管理

企业用户是待测应用的来源，其身份需要经过平台的审核通过方可进行下一步操作。

参与角色	管理员
优先级	高
前置条件	企业用户已注册成功
交互流程	1. 企业用户注册成功，系统发送通知邮件给管理员 2. 管理员进入用户管理页面，查看待审核的企业账号信息，选择审核通过或者不通过。 3. 服务器根据管理员的审核结果发送相应的通知邮件给企业用户。 4. 审核通过的企业用户可以登录系统进行下一步操作
后置条件	企业用户的状态变为审核通过或者未通过

4、测试群主管理

测试群主可以创建测试群并获得管理员的测试应用推送，组织测试群人员进行测试。因其专业性较高，也需要管理员的审核。

参与角色	管理员
优先级	高
前置条件	群主用户已注册成功
交互流程	1. 群主用户注册成功，系统发送通知邮件给管理员 2. 管理员进入用户管理页面，查看待审核的群主账号信息，选择审核通过或者不通过。 3. 服务器根据管理员的审核结果发送相应的通知邮件给群主用户。 4. 审核通过的群主用户可以登录系统进行下一步操作
后置条件	群主用户的状态变为审核通过或者未通过

5、测试群管理

可以将某些测试需求限定在某个测试组内进行，这样测试结果更快更精确。

群主可以创建一个群组来组织一群测试人员，群组的使用需要经过管理员的审核。

参与角色	群主、管理员
优先级	中
前置条件	群主用户已通过审核
交互流程	1. 群主新建一个群组，填写群的基本信息 2. 系统发送通知邮件给管理员，管理员查看新建的待审核的群组，并决定是否通过审核 3. 服务器根据管理员的审核结果发送相应的通知邮件给群主用户。 4. 通过审核的群可以进行下一步操作。
后置条件	数据库中增加一条群组记录并且状态为通过或者未通过

通过审核的群就可以进行使用，用户可以主动加入某个群，只要填写正确的群号以及群口令（线下沟通获得）就可以加入群组。除了用户自己申请加入群，群主还可以主动拉取测试人员进群。

参与角色	测试用户、群主
优先级	中
前置条件	群组存在且状态为审核通过
交互流程	a1. 用户选择加入某个群，填写群号和口令 a2. 系统核对群号和口令，如果正确则将改用户加入到对应群 b1. 群主进入到自己的一个群的管理页面 b2. 群主按姓名搜索测试人员 b3. 系统返回搜索结果列表 b4. 列表不为空的情况下，群主添加测试人员进群 b5. 系统将该测试员添加到对应群组中
后置条件	口令正确则用户加入到该群

用户可以退出某个群，群主也可以剔除群里的某个测试员。

参与角色	测试用户、群主
优先级	中
前置条件	测试用户存在在该群里
交互流程	a1. 用户查看自己的群列表，选择某一个群并退出 a2. 系统将用户移除该群并将操作结果告知用户 b1. 群主进入到自己的一个群的管理页面 b2. 群主将群里的某个测试员退出群 b3. 系统将该测试员移除到对应群组中并将操作结果告知用户

后置条件	测试用户移除该群

6、待测应用管理

测试的第一步就是上次测试应用，这一步由企业用户完成，上传信息完整后将交由管理员审核。

参与角色	企业用户、管理员
优先级	高
前置条件	企业用户已通过审核
交互流程	1. 企业用户创建一个应用并填写基本信息 2. 系统创建该应用并设置其状态为不完整 3. 企业用户上传应用完整信息并保存 4. 系统检测应用的信息是否完整，完整就讲其状态改为待审核 5. 管理员审核应用信息，决定通过和不通过 6. 系统根据审核结果发送邮件通知企业用户
后置条件	系统中存在对应应用，并且状态为审核不通过或者待推送

审核通过的应用，可以被管理员推送到任务广场或者测试群组。

参与角色	管理员
优先级	高
前置条件	测试应用通过审核
交互流程	1. 管理员将通过审核的应用推送到任务广场或者小组 2. 系统生成一条推送记录。应用的状态也随之改变；
后置条件	应用的状态变为已推送

7、发布测试任务

管理员可以对推送到任务广场的应用进行需求分解，创建具体的测试任务；群主可以对推送到自己群里的应用需求进行分解，发布测试任务。

参与角色	管理员、群主
优先级	高
前置条件	测试应用推送到对应群里
交互流程	1. 群主/管理员查看推送到群里的测试应用 2. 群主/管理员创建一个新的测试任务 3. 系统创建测试任务，并反馈操作结果
后置条件	测试任务创建，并且应用的状态为测试中

8、查看/接受测试任务

测试人员可以在网页端或者和移动端上查看当前平台上的测试任务，平台会根据用户的设备类型进行测试任务的定向推送。

参与角色	测试人员
优先级	高
前置条件	测试用户已登录
交互流程	1. 测试员请求查看任务广场或者群组内的任务列表 2. 系统根据用户的设备类型进行任务的筛选并返回结果列表 3. 用户查看某一任务的具体详情并接受任务 4. 系统执行接受任务操作并反馈操作结果
后置条件	系统生成一条接受任务的条目

9、上传/修改测试报告

Kikbug 系统非常重要的一个需求就是用户上传测试报告。用户在客户端上进行测试，测试完成后客户端会将自动收集的信息和用户填写的一些内容上传服务器生成测试报告。测试员可以在网页上查看自己上传的测试报告并修改完善。

参与角色	测试用户
优先级	中
前置条件	测试报告已上传且属于当前操作用户
交互流程	1. 测试用户查看自己接受的任务下上传的测试报告列表 2. 用户点开列表中的某一个报告，查看详情 3. 用户可以修改报告的基本信息并保存 4. 用户可以为报告添加/删改 bug 5. 用户可以上传文档形式的测试报告
后置条件	测试报告的内容更新并能及时查询到最新内容

10、查看测试报告

测试人员可以查看自己上传的测试报告，企业用户可以查看自己上传的应用的测试报告，群主可以查看自己创建的任务的测试报告，管理员可以查看所有的测试报告。

参与角色	测试用户、群主、管理员、企业用户
优先级	高
前置条件	用户对要查看的报告有访问权限
交互流程	1. 测试用户可以查看自己上传的测试报告 2. 群主可以查看自己创建的任务下的测试报告

	3. 管理员可以查看平台中所有的测试报告 4. 企业用户可以查看自己提交的待测应用下的测试报告
后置条件	无

11、测试报告评分

对于测试人员来说，获得测试结果的反馈非常重要。目前，我们提供三种评分，系统自动评分、群主评分、企业用户评分和评价。

参与角色	企业、群主
优先级	中
前置条件	操作用户拥有操作权限
交互流程	1. 企业查看自己上传的应用下的测试报告列表 2. 企业点击列表中的某一个报告查看，并且为其评分和评价 3. 系统记录企业的评分并且将操作结果反馈 4. 群主与企业流程相似，群主可以为自己创建的任务下的报告打分。
后置条件	打分结果记录到数据库并且可以及时查看到

12、待测 app 测试报表查看

对于企业用户来说，希望不要等待最后一刻才看见测试进度是正常的，所以在应用审核通过后，企业用户/管理员时刻都可以查看待测应用的测试报表，包括测试人员数量、报告数量、机型统计、操作系统统计等。

参与角色	管理员、企业用户
优先级	中
前置条件	用户对报表有访问权限
交互流程	1. 企业用户查看自己的待测应用列表 2. 点击某个正在测试中的应用查看详情 3. 详情页中包括了应用的测试情况报表（测试人数、报告数、机型分布、系统分布）和细分报告列表 4. 管理员的流程与之类似，管理员可以查看平台上所有待测应用的测试报表。
后置条件	无

3.2.3 质量需求

Kikbug 系统及其客户端必须满足可用性、负载量、可扩展性、可维护性、安全性、兼容性等质量需求。

1、可用性

网页端和客户端的操作应当简介明了，不应产生歧义，并且为用户在醒目的地方提供帮助页面。

2、负载量

Kikbug 系统将在完成后推广到许多高校，所以用户规模预估会较大，对系统的并发量和负载要求较高。

3、可扩展性

系统能够支持多种客户端，目前应当包括网页端、安卓端和 IOS 端。未来增加新的客户端时要能够开发快速。

4、可维护性

系统的维护不需要修改源代码，至少不需要大规模分散的修改代码，只要更改配置文件就可以做到更改配置，如依赖服务的地址、系统的域名等。

5、安全性

系统必须保证用户的账户信息安全，不会被篡改。内容信息只被符合权限的请求访问，接口不会被非法访问。

6、兼容性

网页端在多种浏览器下的显示相同，功能一致，支持的浏览器最少包含 Chrome、Firefox、Safari、IE9+。

移动客户端在不同的硬件设备上都能运行，功能一致且完整。

3.3 概要设计

如图所示，Kikbug 系统由四部分组成：kikbug 后台接口、网页端、安卓端、IOS 端。安卓端和 IOS 端的用户为测试人员，网页端的用户为所有用户角色，包括测试员、企业用户、群主和管理员。并且 kikbug 系统依赖别的系统提供的服务，包括慕测系统的统一账户中心服务、安全测试中心提供的安全测试服务、自

动测试中心提供的自动测服务以及 upyun 提供的云存储服务等。整个系统的结构与依赖图如图 3.1 所示。

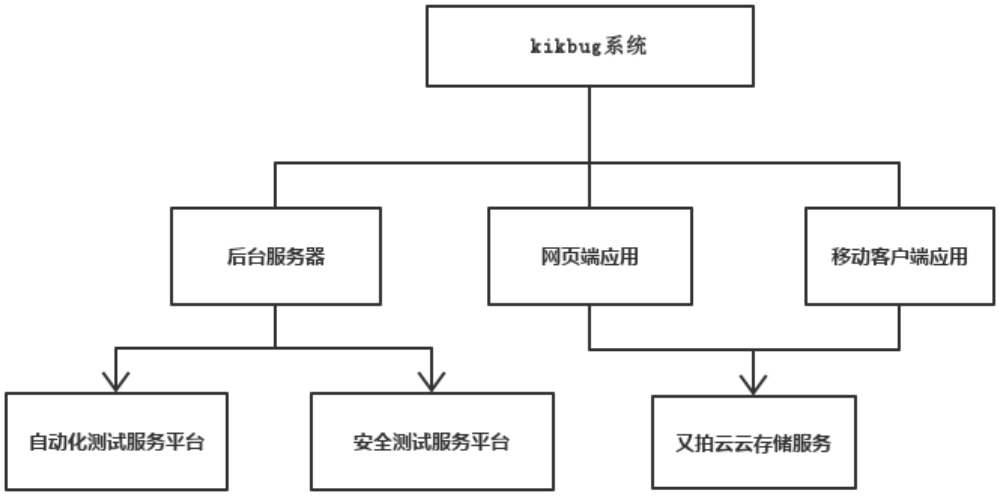

图 3.1 kikbug 系统的结构图

3.3.1 前端架构

Kikbug 的前端项目使用 angularJS 来完成，与传统的 web 项目由服务器端来渲染页面不同，angularJS 使前后端分离由浏览器来发送接口访问请求获得数据并且自己渲染页面，这样可以大大节省服务器端的计算资源和网络带宽。

Kikbug 系统的页面可以按角色分为测试用户类、企业用户类、群主类和管理员类。因为 AngularJS 擅长处理单页面应用程序，因此将整个系统按角色分为四个模块（model），并且将这四个模块中通用的部分抽象出来作为一个父模块供四个子模块继承，减少项目代码量、方便统一修改。模块的结构如图 3.2 所示：

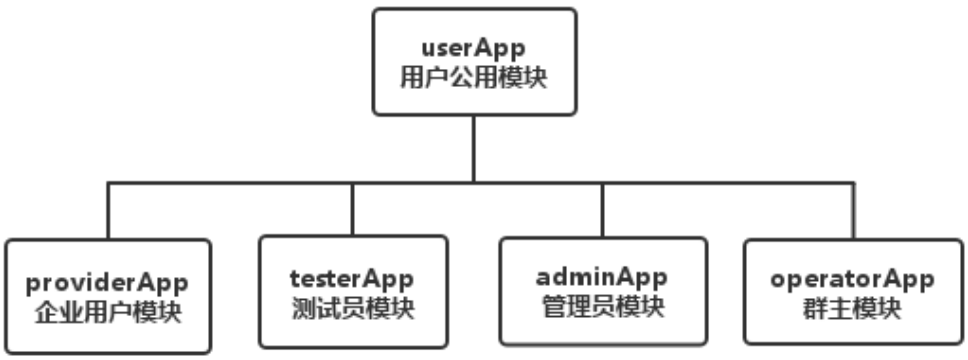

图 3.2 前端结构模块图

Kikbug 前端项目使用 bower 作为包管理器，并且使用 gulp 来进行项目的编译，工作目录存在 app 目录下，编译完的生成用代码存储在 build 目录下。

项目代码的文件结构（app 文件夹内的代码）依照 AngualrJS 的规范进行组织，分为 html、image、js、scss、template 这样几个主要文件夹，如图 3.2 所示。

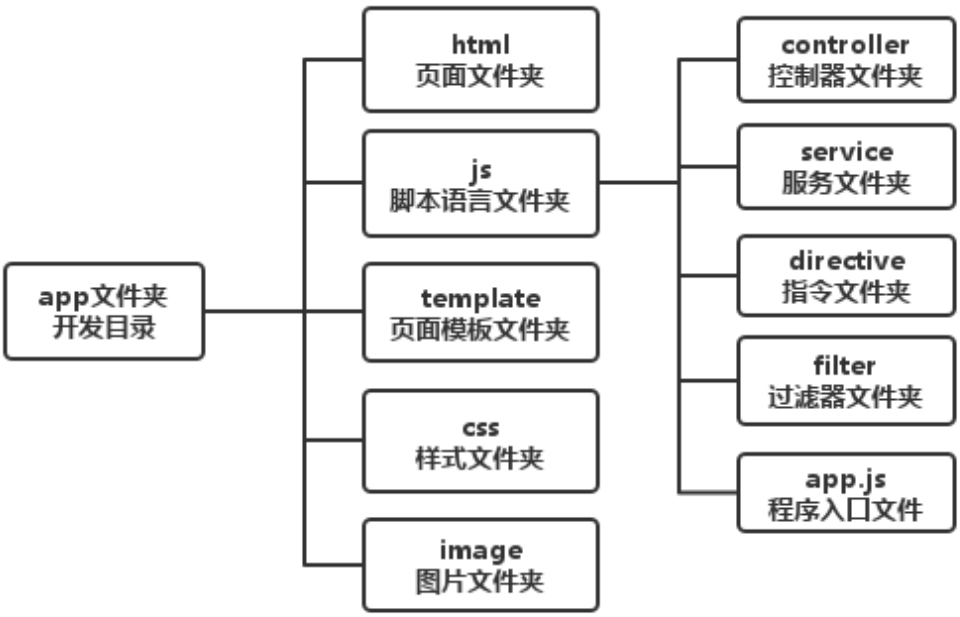

图 3.3 开发目录文件结构图

其中 template 文件夹存放的是页面模板片段，可供主页面重复使用。Js 文件夹存放的是项目最核心的代码，其中的结构如下图所示，其中 app.js 是整个前端项目的入口，里面定义了项目依赖的模块、控制器、服务、过滤器、指令、路由等信息并且包含了项目的配置信息。

前端应用被加载到浏览器以后，渲染 html 页面的同时将开始执行 JavaScript 代码，而这些代码的入口就在 app.js 文件，其执行过程如图 3.4。

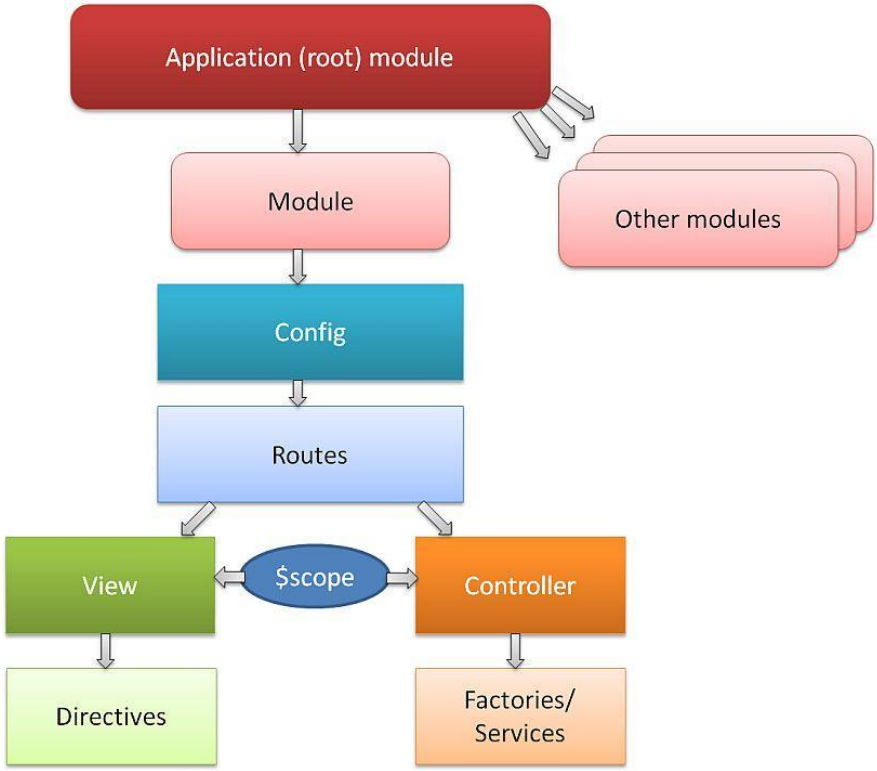

图 3.4 前端脚本代码执行结构图

当浏览器加载一个页面时会根据 ng-model 指令知道这个页面所属的模块（模块的定义在 app.js 中），根据模块配置的路由信息和当前请求中路由地址，浏览器将加载对应的控制器（controller）和模板页面，控制器加载时会执行其中的代码一般会通过网络请求获得数据，并且通过$scope 对象来与模板页面互相传递数据。控制器依赖于服务对象提供的服务，而 html 页面则可以使用 directive 提供的指令来扩展 html 的功能。

举例来说，图 3.5 是 kikbug 首页的截图，整个页面是一个名为 index.html 的页面文件，这个页面的定义模块是 userApp，页面中间红色方框区域部分就是存放在 template 文件夹中的一个模板片段，浏览器根据用户访问的路由地址 /admin 和 userApp 中定义的路由将把对应的控制器和模板文件填充到红色方框区域内。使用路由的好处是可以是页面代码的风格统一并且减少代码量，因为 kikbug 系统的大部分页面都是业务处理页面，整体框架结构是相同的，只是具体业务部分的样式不一样。

图 3.5 kikbug 首页

3.4 本章小结

本章描述了 kikbug 的概述、分析了不同用户角色以及他们各自对应的主要需求和用例描述，并且还从质量要求的角度去分析系统的质量需求，包括可用性、负载量等。

然后本章简要的介绍了 kikbug 系统的架构设计，详细的介绍了 Kikbug 网页端应用的架构设计。

第四章 详细设计与实现

4.1 前端模块概述

Kikbug 网页端系统按功能分，主要分为 controller（控制器）、service（服务）、filter（过滤器）、directive（指令）这几个部分。

1、Controller（控制器）

Controller 是常规的 JavaScript 对象，由标准的 JavaScript 对象的构造函数创建。它是前端 JavaScript 代码中最重要的部分，它由 ng-controller 指令指定、能控制应用程序的数据，是实现业务逻辑的主要部分。其具体实现如图 4.1 所示。

```
module.exports = function ($scope, sweet, userService) {
    //定义一些变量绑定在$scope 对象
    $scope.isUser = true;
    $scope.errMsg = "";
    $scope.login= function (isValid) {
        //使用 userService 提供的登陆方法进行用户登陆
    userService.login($scope.user.id, $scope.user.password, $scope.user.role)
            .success(function (response, status, headers, config) {
                ...//登陆成功后做的事情省略
             })
            .error(function (response, status, headers, config) {
                    sweet.show('', '登陆失败!请检查用户名和密码!', 'error');
            });
    };
};
```

图 4.1 测试用户登录 controller 代码缩略图

2、Service（服务）

在 AngularJS 中，服务是一个对象或者函数，可以在你的 AngularJS 应用中使用。AngularJS 提供了 30 多个服务，例如$location 服务，它可以返回或者修改当前的 url 地址。

使用服务的好处是当你在多个地方使用到相同的代码时，你可以将这部分代码抽象成一个服务，减少代码量并且易于修改。Kikbug 的前端系统提供了功能丰富的服务供 controller 使用，其中最重要的一块就是网络请求部分，虽然 angularJS 已经提供了$http 服务供应用发送网络请求，但是在控制器中直接使用$http 服务还是会导致代码量大且重复，所以我们提供了针对业务的网络请求

封装。首先是 connector 服务，connect 服务封装了后端服务的访问地址以及最基础的 get、post、put、delete 等请求，如图 4.2 所示：

```
var baseUrl = 'http://localhost:8080/kikbug-api';
this.$get = function($http) {
    var session = '';  //session 值是登陆后服务器端返回的，需要在每次请求的时候带上
    return {
        setSession: function(s) {….},  //设置 session 的方法
        //get 请求的封装
        get: function(path, params) {
            return $http({
                method: 'GET',
                headers: {'Session': session},
                params: params,
                url: baseUrl + path,
                responseType: 'json',
                cache: false
            });
        },
        //post、put、delete 请求的封装，略
        post: function(path, params) {…},
        put: function(path, params) {…},
        delete: function(path, params) {…}
    };
};
```

图 4.2 网络服务 connector 实现代码缩略图

在 connctor 服务的基础上继续封装与业务相关的网络请求，如与任务相关的请求：搜索任务、查询任务详情、接受任务、创建任务等，都封装在 taskService 服务中（taskService 依赖 connctor 服务），控制器可直接调用这些服务，服务的实现如图 4.3 所示，使用在图 4.1 的登陆控制器中可以看见。

```
module.exports = function(connector) {    //传入 connector
    var basePath = '/task';      //同类请求公用的基础 url
    //搜索任务的方法封装
    this.search = function(userId, appId, groupId, page, count) {
        var path = basePath;
        var params = {
            userId: userId,
            appId: appId,
            ……
        };
        return connector.get(path, params);  //调用 conntector 的 get 请求方法
    };
```

```
    this.create = function(task) {……};
    this.get = function(taskId) {.....};
    this.hasTask = function(taskId) {.....};
};
```

图 4.3 taskService 的实现代码缩略图

除了为网络请求封装了服务外，kikbug 还封装了其他许多服务，这里列举一个跟请求安全比较相关的服务 cookieService。因为 kikbug 系统的架构设计是前后端分离，所以后端只提供 restful 的接口，这样接口访问的权限必须受到严格控制，否则接口的安全会受到威胁。Kikbug 的做法会过滤请求的 header，只有是来自 kikbug.net 域名的请求才会被接受，并且是在用户登录成功的时候，系统为该登录账户自动生成一个有时限的 session，并且返回给调用者，接下来的所有请求调用者都必须在请求头里带上这个 session。网页端登录成功后会将用户的 id、角色、session 都存储到 cookie 当中，并且在发送请求的时候自动把 session 值放入请求头当中，如果 cookie 中找不到这个 session，那么页面就会自动跳到登录页，让用户进行登录操作。cookieService 的具体实现如图所示：

```
module.exports = function($cookieStore, propertyService){
    var prefix = 'kikbug-';       //cookie 存储前缀
    this.setUserInfo = function(role,info) {    //存储 cookie
        $cookieStore.put(prefix + role + '-userId', info.userId);
        $cookieStore.put(prefix + role + '-session', info.session);
    };
    this.getUserInfo = function(role) {    //读取 cookie
        if (!$cookieStore.get(prefix + role + '-userId')) {
            //如果 cookie 读取不到就跳转到登陆页
            window.location.href = propertyService.getProperty('domain') + '/';
        } else {
            var info = {};
            info.userId = $cookieStore.get(prefix+role + '-userId');
            info.session = $cookieStore.get(prefix+role + '-session');
            return info;
        }
    };
    this.clear = function(role) {……};
};
```

图 4.4 cookieService 的实现代码缩略图

3、Filter（过滤器）

过滤器可通过一个管道字符（|）和一个过滤器添加到表达式中，这一功能可以使得页面内容的显示更加定制化、更加方便。AngualrJS 内置了许多方便使用的服务，如货币格式化过滤器（currency）、数组排列顺序过滤器（orderBy）等。

Kikbug 前端系统也提供了自己的过滤器，比如应用的状态后端接口返回的值为 int 类型，如果直接显示在页面上肯定会让用户感到迷惑，所以提供一个关于应用状态的过滤器，根据 int 值显示中文状态并显示为不同的颜色，其实现如图 4.5 所示：

```
//状态中文显示过滤
app.filter('appStatus', function () {
    return function (value, isAdmin, distributionStatus) {
        if (value === 0) {
            return '信息不完整';
        }
        ……
        } else {
            return '火热测试中';
        }
    };
});
//状态表示颜色过滤
app.filter('appStatusColor', function() {
    return function(value, isAdmin) {
        if (value === 0) {
            return '#3484de';
        }
        ……
        } else {
            return '#f1842e';
        }
    };
});
```

图 4.5 应用状态过滤器实现代码缩略图

4、Directive（指令）

AngularJS 不建议用户像传统的 JavaScript 代码那样直接在 JS 代码中操作页面 dom 元素，而是使用通过被称为“指令”的新属性来扩展 HTML，如 ng-app、ng-controller、ng-repeat 等都是 angular 内置的指令，其中 ng-repeat 可以重复一个 html 元素。

Kikbug 系统根据需求定制了自己的指令，如页面侧边栏的悬浮组件就是使用自定义的指令生成，实现代码如图 4.6 所示：

```
module.exports = function() {
    return {
        scope: {
            scrollTarget: '@'       //向上滚动到顶端的页面元素
```

```
        },
        templateUrl: 'templates/piece/float-panel.html',    //侧边栏的页面文件
        restrict: 'E',
        controller: function($scope) {
            $scope.scrollTop = function() {          //滚动方法
                $($scope.scrollTarget).scrollTop(0);
            };
        }
    };
};
```

图 4.6 悬浮组件指令实现缩略图

4.2 公共界面的设计与实现

kikbug 前端页面按角色分大致分为五部分，其中四种用户角色各占一部分，另外大家都通用的页面，登录注册等公共页面单独为一部分。下面就来介绍一下这部分页面的设计与实现。

这部分的页面使用 index.html 作为父页面，userApp 作为顶端模块，根据 userApp 中配置的路由表来切换中间的显示区域。共有以下几个子页面，普通类用户登录界面、管理类用户登录界面、测试员注册页面、企业用户和群主申请注册页面和注册页面。

因为这部分页面的实现难度不是很大，也没有很多特别的实现点，这里就不一一赘述了，就只展示一下测试用户的注册页面实现。Kikbug 的测试用户规定只能使用手机号注册，且必须验证手机验证码。这里实现中值得注意的是，表单验证没有使用传统的方式，而是使用了 angular 提供的表单验证指令，例如手机号的验证，不需要你在 js 代码中验证是否已经填写格式是否正确，你只要像下面这样写：

```
<input class="form-control" name="mobile" ng-model="tester.mobile" required ng-pattern="/(^1[0-9]{10}$)/">
```

angular 的表单验证就会自动帮你验证手机号是否已经填写，格式是否正确。只有当所有表单内容都符合你给的规定，下面的注册按钮才会变为可点击状态，非常的方便。登陆页面的界面如图 4.7 所示。

图 4.7 测试用户登录界面截图

4.3 管理员界面的设计与实现

4.3.1 界面设计

管理员是 kikbug 系统中一个重要的角色，它的行为有很多，包括了运营人员的职责。管理员的界面分为这样几个部分，应用管理、推送管理、任务管理、群组管理、用户管理和客户端管理，这些界面部分的入口在侧边栏都可以找到。

“应用管理”部分页面主要是向管理员显示系统当前提测的应用列表以及应用的详细信息。提测的应用有以下几个状态：新建不完整态、内容完整待审核态、审核通过态、已分解态。另外还有一个推送状态来表示当前应用的推送情况，因为应用可能不仅仅只推送到一个地方，比如同时推送到任务广场和自动测试，所以推送状态不是使用单一值来表示而是使用位值来表示，状态值按低位开始每一位分别表示推送到任务广场、小组、自动测试、安全测试。管理员也有权限可以修改应用的部分信息，所以当应用处于审核通过之前，详细内容页面上保留有修改应用内容的入口。

“推送管理”部分页面是用来想管理员展示应用的所有推送列表和推送的详情。推送主要分两类，一类是推送到任务广场的应用（不可以再推到小组），另一类是推送到测试小组的应用（可推送至多个小组但不能再推到任务广场）。对于推送到任务广场的应用，管理员可以为其创建具体任务也可以看见已有的任务列表，但是对于推送到测试小组的应用，管理员只可以看见该应用的测试报告列表和总结报告而具体划分的任务管理员不可见。

“任务管理”是一个快速入口，可以让管理员快速查看任务广场上的任务列表及其详细信息。详细信息页上可以看见任务的具体信息和当前任务已上传的报

告列表。

“群组管理”页面帮助管理员管理当前系统中存在的群组，群主新创建的组群会在这里展示出来，管理员可以查看群组详情并且决定是否通过审核。

“用户管理”部分页面是帮助管理员对系统中新注册的企业用户和群主进行审核的。在这里可以查询待审核、审核通过和未通过的用户并查看他们的具体信息以决定是否要通过审核。

“客户端管理”页面是为了方便管理员对客户端的版本信息进行管理，因为客户端可能会进行多次迭代更新，而手动更新下载链接涉及到重新部署项目的问题，所以提供管理页面可以上传新的客户端而链接会自动更新。

4.3.2 关键页面实现

导航栏的实现

这里说一下侧边导航栏的实现，管理员的主页面同之前的公共页面设计一样，也是使用一个父页面加路由切换子页面来实现的（子页面就是图 4.8 中红色方框内圈出来的部分）。

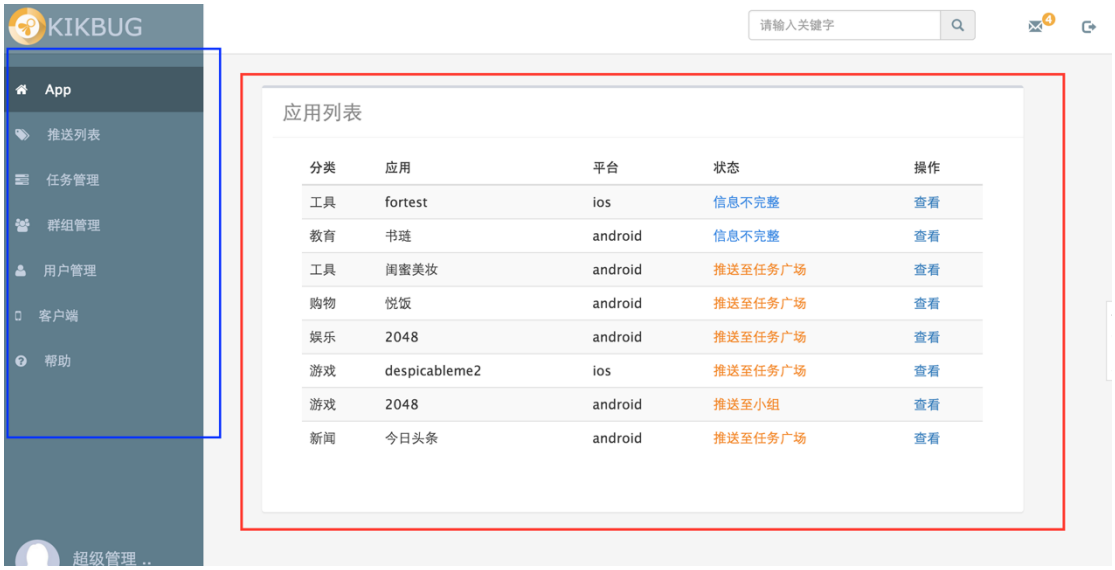

图 4.8 管理员主页面

为了使页面实现高度的可配置化，导航栏的条目没有写死在代码中，而是写在一个 json 文件中，在每次加载页面的时候自动获取到导航条目并且显示出来，而导航条目的活跃属性使用的是一个监听器完成的，每当 url 地址有改变的时候监听器就会被触发，检测当前的 url 匹配的导航项是哪一个，并将其的 active 属性设置为 true。这个 json 文件格式如下：

```
[{"name": "App","icon": "home","url": "/apps"},..]
```

加载的方式是在一个父 controller 中实现的即 rootController，所有的页面都

会由这个 controller 来控制，这个 controller 会在加载的时候获取到你当前的用户角色然后去读取相应的 json 文件，并将其内容解析后存储在 $scope.menuItems 中，然后在 html 中使用 ng-repeat 来依次显示。在解析 json 文件的同时，也会初始一个变量和$scope.menuStates，他记录了每个 url 是否当前正在访问的 url，如果值为 active 的话，那么在导航栏上就会加重显示出来。而监听器做的工作就是这样，当 url 变化时，将$scope.menuStates 进行遍历，看其是否为当前访问的 url，如果是的话就把它的值设为 active，否则为空，这样就能很好的处理导航栏的活跃状态了。

应用管理

图展示了应用处于测试者中的页面。在页面中可以看见应用的测试状态统计报表，包括测试员的人数、报告数量、机型统计、系统统计信息，如图 4.9 所示。

图 4.9 统计结果报表截图

图标使用的绘制插件是 angular 定制的 chart.js，这个插件在 chart.js 的基

础上进行了封装，只要在 html 页面上使用一个简单的指令，告诉指令图标的类型、数据和标签的数据源，这个指令就会自动绘制图表。这里关键的实现就是如何将服务器端返回的数据转换为指令使用的数据格式，服务器端的统计实现是对于测试人数和测试报告数提供单独字段进行存储而机型和操作系统的统计则使用 json 字符串存储，统计方式是实时统计，没当一个统计事件发生时，系统就抛出一个事件然后由专门的统计线程捕获进行统计数据的更新。而客户端向服务器发送的统计数据访问请求得到的数据就是一个 json 对象，对这个 json 对象中的机型数据（也是 json 对象）进行遍历，然后生成两个数组 data 和 labels，然后存储在 mobileModel 对象中，这样指令中就可以拿来使用了。

群组和用户管理

管理员对群组和用户的审核页面如图 4.10 所示，页面上提供了筛选项，当筛选条件改变时页面的显示列表也会做相应的改变。

图 4.10 用户管理和群组管理界面

实现方式是通过给下拉框添加 onChange 监听器，当下拉筛选框的内容发生改变时监听器就会被触发，程序会根据筛选条件向服务器发送对应的请求，然后将数据对象进行更新，但是不需要对页面的 dom 元素进行操作，因为 angular

是通过$scope 对象进行双向数据绑定的，当 controller 对$scope 上绑定的数据对象做改动时，页面也会对应的进行更新。这里有个小问题，就是下拉筛选框绑定的数据对象本来按照双向绑定的原则当页面发生改变时，controller 中获得的对象也会随之改变，但实际操作时却不会实时改变，需要在调用监听函数的时候手动把下拉框的值传入。

客户端管理

管理员上传客户端的页面如图中蓝色区域所示，因为 kikbug 的服务器资源有限，为了满足高负载高流量的要求，kikbug 上的所有文件都存储在 upyun 上，所以如果不提供上传更新页面的话，管理员自己传到 upyun 然后再更新链接是一件非常麻烦的事情。现在提供了更新页面，处理上传到云端的细节并且将新的链接存储在服务器端，当用户请求访问下载链接的时候读取（服务器端对这个接口有做缓存，所以速度可以保证），并且是用在线自动生成二维码的技术实时生成二维码，方便用户下载，如图 4.11 中红色区域所示：

图 4.11 客户端上传和下载界面

客户端上传到 upyun 使用的是 kikbug 自封装的 uploadService 提供的服务，该服务将 upyun 的上传配置暴露出来，使用的时候只需要在 config 中以 json 的格式配置一下就可以了，配置格式如下：

```
upyunProvider.config({
  form_api_secret: '52TA3sfwGKHqsHO..',
  bucket: 'kikbug-..',
  'allow-file-type':'jpg,png,txt,doc,docx,pdf,apk,ipa',
  'save-key':''/{year}/{mon}/{day}/upload_{filemd5}{.suffix}'',
..
});
```

uploadService 提供一个接口方法，可以让调用者上传文件，调用者只需要传入上传文件的 form 和上传结束后的回调方法，uploadService 就可以自动上传表单中的文件，uploadService 的具体实现在图 4.12 中可见。

```
module.exports = function(upyun,sweet) {
    var listeners = [];
    upyun.on('uploading', function(progress) {
        if (listeners.length < 1) return;
        var activeListener = listeners[listeners.length - 1];
        activeListener(progress);
    });
    this.upload = function(form, cb) {
        console.log('正在开始上传...');
        var file = {};
        upyun.upload(form, function(err, response, file) {
            if (err) console.error(err);
            console.log('返回信息：');
            console.log(response);
            console.log('文件信息');
            console.log(file);
            if (file.code === 200 && file.message === 'ok') {
                sweet.show('上传成功','','success');
                cb(file.absUrl,file.file_size);
            } else {
                sweet.show('上传失败','','error');
            }
        });
    };
    this.pushListener = function(listener) {
        listeners.push(listener);
    };
    this.popListener = function() {
        listeners.pop();
    };
};
```

图 4.12 uploadService 实现代码缩略图

有时候上传的文件较大，时间较长，用户可能希望知道当前上传的进度，所

以 uploadService 还提供了进度监听器，只需要调用 pushListener()方法，将监听器传入，service 会在每次进度有更新的时候都将当前进度值传给监听器，监听器拿到数据后就可以在页面上显示出来，kikbug 当中使用了 bootstrap 提供的进度条来显示上次进度，效果如图 4.13 所示，

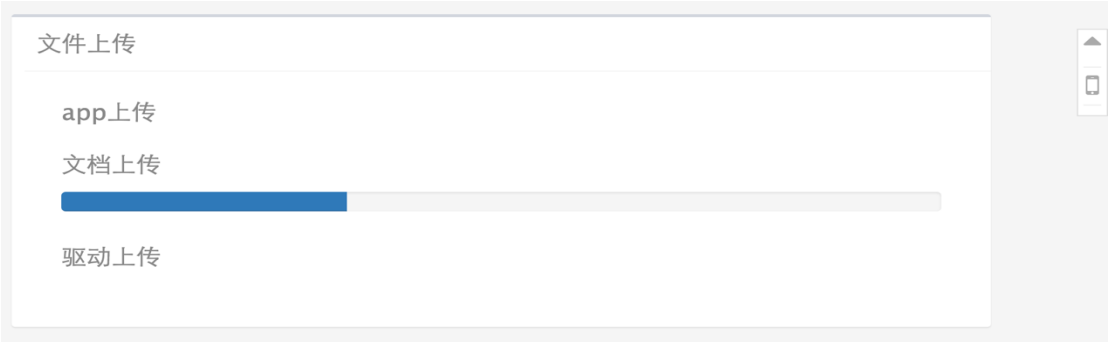

图 4.13 上传文件进度显示条

4.4 企业用户界面的设计与实现

4.4.1 界面设计

企业用户的页面设计很简单，就是管理我的待测应用、查看测试结果和查看修改个人信息这三样。下面来分别介绍一下这几块的内容分布。

企业用户登录以后首页显示的就是我的待测应用列表并在最后给出添加新的待测应用的入口。点进待测应用的详情页面可以看见应用的测试情况包括统计报表、测试报告和总结报告，这和管理员查看到的应用详情页面内容类似。

“测试结果”页面如同上文所说，有一部分是包含在了应用详情页当中的，详情页中有测试报告的列表，点击进去可以查看报告的详细内容。并且还可以给该报告的内容进行打分和评价。

“个人信息”子页面显示了企业用户的介绍信息，包括联系方式地址什么的，这些都可以进行修改。

4.4.2 关键页面的实现

待测应用管理

待测应用管理包括查看和添加这两类操作，其中查看的实现难度不大，就是向服务器发送查询请求然后将结果实现就可以了。这里来重点阐述一下添加的实现，因为一个待测应用的内容包括了较多的内容，除了应用的基本信息外，还需

要上传 apk/ipa 文件、测试需求文档、应用驱动等信息，因此如果让用户一下子就把信息填写完整然后上传到服务器端是有难度的，因为用户可能在上传某个文件的中途去做了别的事情，然后页面又被不小心关闭了，那么用户就需要重头开始做刚刚做好的那些事情。

所以在这里我们的设计是这样的，用户点击添加新的应用，然后跳出弹出框，弹出框的样子如图 4.14 所示，让用户填写新应用的一些基本信息（不涉及大的文件的上传），填写后点击创建，这时候前端程序会将这些基本信息组织起来上传到服务器端生成一个内容不完整的测试应用，然后页面跳转到该测试应用的详情页，用户可以在这个时候分别上传 apk/ipa 文件、测试需求文档等文件，每次上传成功服务器端都会实时更新。当所有必须的文件都上传结束后，应用的状态会变为待审核状态。

图 4.14 应用创建模态框

这里展示一下修改 app 信息的实现代码，因为从服务器端获得的应用信息字段的格式不能直接用于页面展示，所以在 controller 中需要对这些字段的内容进行解析，然后在保存修改的时候再讲解析出来的字段内容重新拼接成服务器端接受的数据格式。这里以测试方式为例做一下介绍，应用的测试方式有三种可选，用户可以选择最少一种最多全部，服务器端为了存储方便就将这个选择按位存储在一个 short 数值中，controller 拿到这个值以后对其进行位运算求出这几种测试方法有没有被选择，保存的时候按位生成这个值。具体实现的代码如图 4.15 所示：

```
$scope.getAppInfo = function () {
    appService.get(appId)
```

```
            .success(function (response, status, headers, config) {
                $scope.app = JSON.parse(response.data);
                //对测试方法的解析
                $scope.app.isManual = ($scope.app.taskType & 1) !== 0 ? true : false;
                $scope.app.isAuto = ($scope.app.taskType & (1 << 1)) !== 0 ? true : false;
                $scope.app.isSecurity = ($scope.app.taskType & (1 << 2)) !== 0 ? true : false;
                ……
            })
            ……
};
//创建应用时对测试方法字段的计算
$scope.createApp = function (category, platform) {
    var taskType = 0;
    if ($scope.type.manual === true) {
        taskType = taskType | 1;
    }
    if ($scope.type.auto === true) {
        taskType = taskType | (1 << 1);
    }
    if ($scope.type.security === true) {
        taskType = taskType | (1 << 2);
    }
……
}
```

图 4.15 应用创建修改代码缩略图

测试结果查看

企业用户在应用详情页里可以看见测试的结果报表、总结报告、细分报告，页面的结构和样式与小节 4.3.2 中的图类似，这里就不赘述了。现在主要来讲讲测试报告的组成和实现，如图 4.16 所示，可以看到测试员上传的测试报告内容包括以下几个部分，基础信息、系统日志、步骤描述、bug 列表和详细报告。

图 4.16 测试报告展示页面

对于测试员上传的测试报告，企业用户有必要给予适当的反馈，点击修改评价，打分和评价条目就都变成可编辑的了，修改评分和评价后再点击保存就可以了。这里来说一下这个显示打分的组件，为了不直接显示打分的数字，我们使用了一个叫 angular-input-star 的插件，这个插件提供一个指令可以直接在 html 页面上使用，只要绑定到 controller 中的一个整数值就可以了，具体使用方法如下：

```
<input-stars max="5"
ng-model="report.operatorScore"readonly> </input-stars>
```

4.5 测试用户界面的设计与实现

4.5.1 界面设计

测试员是整个 kikbug 系统中的支柱，kikbug 的正常运转离不开测试员的参与，所以在设计测试用户界面的时候考虑的是简洁明了，目的是让用户使用方便将重心放在测试上，并且方便的上传报告。测试人员的最简单的流程就是查看任务广场、接受任务、完成测试、上传报告、完善报告，其中完成测试和上传报告不在网页端进行而在移动端进行。

“任务广场”子页面就处于测试员的首页，测试员一登录系统就能看见任务广场页面。任务广场中显示正在测试中的任务，按结束时间进行倒叙排列，并且在右侧显示当前系统中的积分榜。

“我的群组”部分页面显示的是我也加入的群组列表，并含有加入新群组的入口。点击某个群组就可以看见群组内的详细信息，包括群组基本信息、群组内的成员、群组内的任务。

“我的任务”页面包含的是测试员已经接受的任务列表页，如果还没有接受任何任务，那么系统会提醒你去任务广场看看。点进任务详情看的话可以看见任务的详细信息，包括任务来源，描述，我上传的报告列表。

“个人中心”页面就是让用户查看修改自己的个人信息的地方，用户可以在这里查看自己的积分、信息，并且修改密码、个人信息。

4.5.2 关键页面的实现

任务广场

任务广场是用户一登录就会看见的页面，所以设计的风格要明亮整洁清晰，我们查看了大量同类型的网站的任务广场页面，最终决定以扁平化和响应式的设计风格来实现页面，我们现在的页面可以按浏览器窗口的大小不同来动态的显示，能较好的适应不同设备上的浏览器，如图所示，是浏览器正常大小时候的样子，当浏览器的宽度变窄时，页面的布局方式也会随之进行改变，如图 4.17 和图 4.18 所示。

图 4.17 浏览器页面大小正常时的页面布

图 4.18 浏览器宽度发生变化时

这种响应式设计除了使用 boostrap 提供的栅格系统来实现页面上部分区域的响应式问题，我们还借鉴了 bootstrap 对于不同设备大小的定义，实现了我们自己的响应式布局类（boostrap 的设备大小定义如表格所示）。

表格 4.1 设备屏幕大小定义表

设备	超小设备手机	小型设备平板	中型设备	大型设备
屏宽	<768px	>=768px	>=992px	>=1200px

Class 前缀	.col-xs-	.col-sm-	.col-md-	.col-lg

我们的响应式布局类是使用@media 查询来实现的，侧边导航栏是包含在一个 menu-side 的 class 中的，我们对这个 class 的占用宽度在不同设备大小上做了一次定制，我们将屏幕划分成两类，一类是小型设备（<992px），另一类是中大型设备（>=992px），并且给导航栏上的文字加上 hidden-xs、hidden-sm 的 class，使其在导航栏变窄的时候不显示。

```
<span class="hidden-xs hidden-sm">{{ item.name }}</span>
```

任务列表和排行榜的布局也是使用了 bootstrap 的栅格系统，定义了在不同屏幕大小时候组件占用的列数（一行总共有 12 列），这样随着屏幕大小的变化，任务列表和排行榜总能占据适当的宽度来显示自己。

这里的任务列表的展示内容经过筛选只展示了比较重要的一部分，有应用图片能让人一眼就看明白这个应用是什么，还有任务名、应用分类、截止时间、积分等信息。其中任务详情的显示为了占用的长度不超出设定的最大值而使页面风格不一致，我们写了一个过滤器来过滤字符串的长度，给定过滤器一个最大长度，当字符串的长度超过这个最大长度的时候，超出的部分就会被替代字符“..”给代替。当一个任务你已经接受过了的时候，任务的右下方还会显示已接受字样。

测试报告

对于测试用户来说，另一个非常重要的功能就是测试报告的修改和补充，因为在移动端上打字的不方便，所以在网页端上进行细节的描述就非常重要。我们设计的测试报告包括这样几个部分：基础信息、系统日志、步骤描述、bug 列表和详细报告。报告的显示同 4.4 节企业用户界面设计与实现中的图类似。

其中基础信息都是由客户端自动收集上传的，有手机的品牌、型号、操作系统、测试用时、创建时间。系统日志也是由手机客户端自动收集的，包括用户的点击事件、窗口改变、出现弹窗、内容改变等等，格式下所示，其中包括事件类型、时间、包名、动作等信息，记录这个日志的目的就是为了 kikbug 的后续工作，根据日志还原测试过程服务的。

```
EventType:TYPE_VIEW_TEXT_SELECTION_CHANGED;
EventTime:90334683;PackageName:com.func.kikbug;
MovementGranularity: 0;
Action:0[ClassName:android.widget.EditText;Text:[测试];
ContentDescription: null; ItemCount: 2;
……
; OutBounds: Rect(178, 346 - 1184, 442); SyscTime: 1460589987121
```

步骤描述可以选填，一般来说测试之前都会设计测试用例，设计的测试用例就可以填写在步骤描述当中。

测试报告中最重要的部分就是 bug 列表了，对于每个 bug 用户可以写一段话描述一下，并且选择严重程度，再上传九张以内的图片。对于 bug，用户可以在手机端上创建，然后在网页端上修改，也可以直接在网页端上修改。点击修改或者创建按钮，就会跳出一个模态框让用户来填写 bug 的具体信息。具体页面如 4.19 所示。

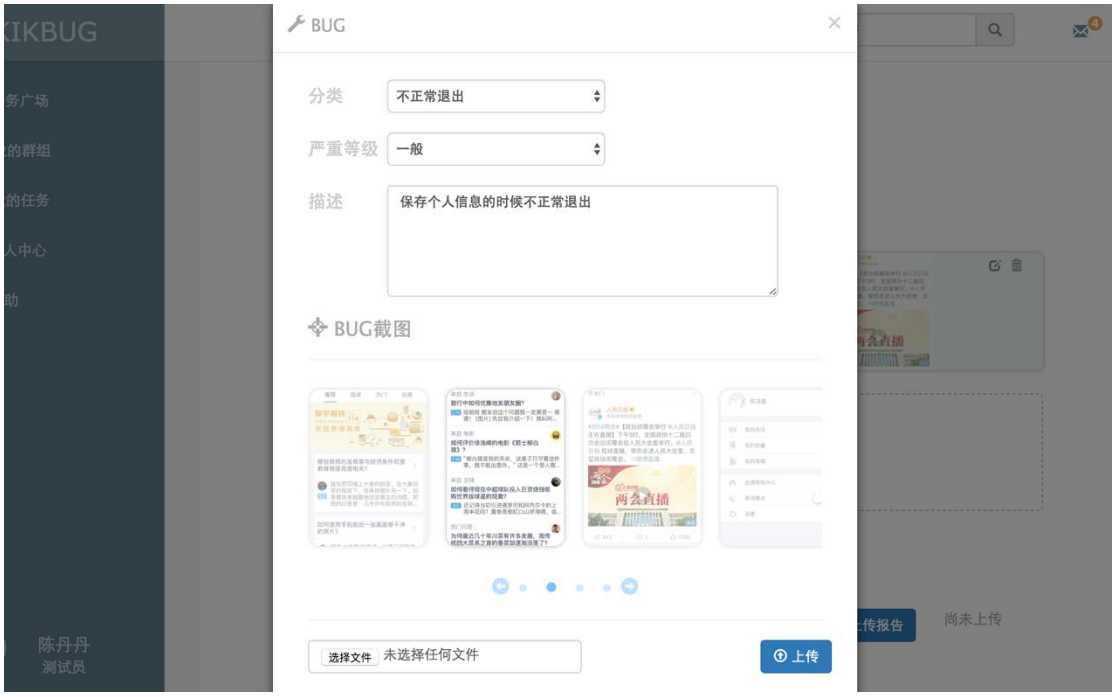

图 4.19 bug 创建和修改模态框

这里来讲一下模态框的实现，bootstrap 提供了一个 angular 版的插件 angular.ui，使用这个插件提供的服务$uibModal. open()可以直接打开一个模态框。因为模态框其实也是一个 html 页面片段，所以这个 open 方法接受一个对象参数，对象中定义用来显示的 html 模板页面、控制模板页面内容的 controller 以及一个 resolve 对象，该对象内定义一系列数据和方法作为 controller 的初始化参数。Bug 的图片最多可以上传九张，为了存储的方便这些图片的地址被以“;”为间隔符拼接组成。当打开模态框的时候这个图片地址字符串被解析为一个数组，对这些图片的操作会反应在数组中，当用户修改完点击保存的时候，数组会被重新合并成字符串然后更新到服务器中。

最后，有些用户可能觉得这样的结构化的报告不能很好的表达自己的测试内容，那么也可以自己编写一份文档并且上传到详细报告中。

4.6 群主界面的设计与实现

4.6.1 界面设计

群主角色是第二版的 kikbug 新增的角色，主要是为了让线下有交流的一群测试员组织在一起进行有针对性的测试。群主的页面组成不太复杂主要有推送管理页面和群组管理页面。

“推送管理”部分页面是群主登陆后见到的主页面，在这里群主可以看见管理员推送过来的待测应用（推送到我所有的群里的应用都会在这里显示），一目了然。点进去就是推送应用的详情页，在这里可以看见详细信息还可以创建任务、上传总结报告、查看测试报告列表。

“群组管理”部分页面的入口就在导航上的我的群组，点击进去可以看见我已有的群组和新建群组的入口，点击某个群组可进入群组详情页，详情页内包括群组信息、群组成员和推送到该群组的应用列表。

4.6.2 关键页面的实现

群组管理

群组管理对于群主来说是一个比较重要的事情，kikbug 提供的群一般为线下有交流的群，群主创建群的时候会给定一个口令，只有知道群号和群口令才能加入一个。群主的群组列表页设计的时候采用了扁平化的设计风格，页面风格如图 4.20 所示。

图 4.20 群组列表页

可以看见每个群的第一个字被做成了 logo，并且底色不一样，这是使用一个叫做 color 的原生插件实现的，使用的方法是，先将群组的名字使用 md5 进行一下编码传入插件提供的构造函数中创建出一个 color 对象，然后依次调用该对象 saturation(float)、lightness(float)、hex()，方法来调节颜色的饱和度、亮度、

获得十六进制的颜色编码。当然这个插件因为是原生的所以不能直接被页面使用，我们将它的功能封装到一个名为 color 的过滤器当中了，使用的时候只需要传入群组名和亮度（可选、默认值为 0.5），过滤器就会返回一个颜色值，使用具体方式如下所示。

```
<div class="circle" style="background-color:
{{ group.name | color }};">
```

群主在群组详情页里可以进行组内成员的管理，可以移除某个成员，也可以添加某个成员。移除的时候为了用户避免误操作，系统都会弹出确认框，只有经过确认以后系统才会执行移除操作，界面的显示如图 4.21 所示：

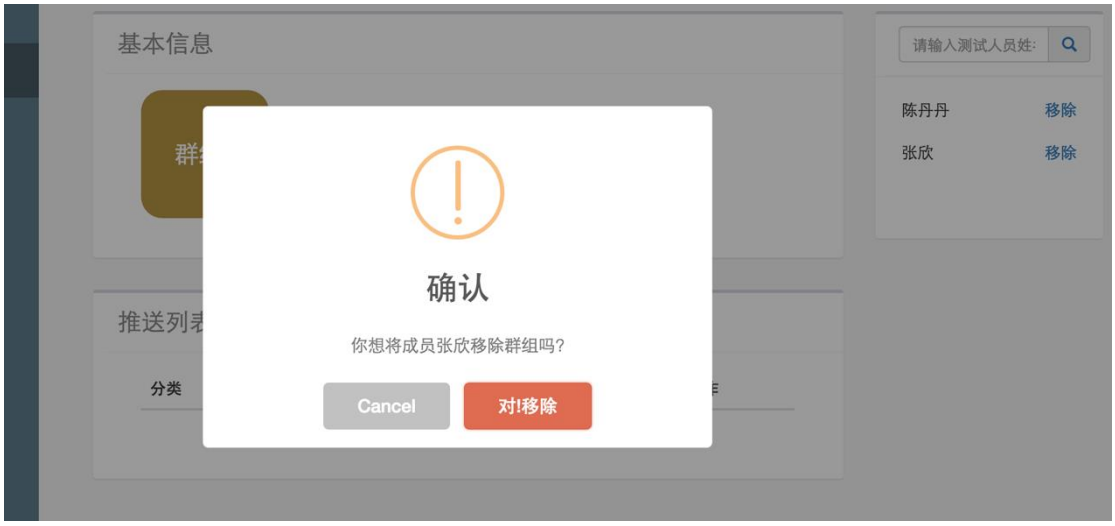

图 4.21 移除组内成员提示

添加小组成员的成员的方法与之类似，在图中成员列表上方的搜索框内输入要添加的测试员的姓名，然后页面上回显示搜索结果，点击添加就可以了，当然这里为了确保用户不是误操作，也会有类似的提示框弹出。采用搜索然后再添加的方式，可以方便用户在不记得测试员的准确姓名的时候依然能找到想找的测试员。

这里来讲讲 kikbug 使用的弹出框，本来 kikbug 系统使用的弹出框是传统的浏览器自带的对话窗口，但是传统的 js 弹出框的样式不太美观也和整个系统的页面风格不相匹配，所以经过筛选我们选用了一款非常热门的弹出框插件。这款插件的样式非常丰富、除了提供默认的样式，你还可以自己写一个 html 片段用来弹出。这款插件 sweet-alert 的使用也非常方便，只需要调用它提供的 show() 方法，并且传入显示参数就可以了。这里截取一段添加组员的弹出框代码作为参考，代码见图 4.22。

```
$scope.addMember = function (testerId,testerName) {
    sweet.show({
```

```
            title: '确认',
            text: '你想将测试员' + testerName + '加入群组吗?',
            type: 'warning',
            showCancelButton: true,
            confirmButtonColor: '#DD6B55',
            confirmButtonText: '对!加入',
            closeOnConfirm: false
        }, function () {
            sweet.show('', '即将添加成员' + testerName, 'success');
            groupService.joinGroup(testerId, groupId, "operator").success(function (response) {
                $scope.getGroupMember();
                $scope.testerList = null;
            }).error(function () {
                $scope.testerList = null;
                sweet.show('', '添加成员失败!', 'error');
            });
        });
};
```

图 4.22 弹出框实现代码

可以从上述代码中看到，sweet.show()方法接受两个参数，第一个参数是个对象是将要显示在弹出框上的内容，第二个参数是一个方法，这个方法将在用户点击确认后执行。

推送管理

群主的另一个非常重要的事情就是要管理管理员推送过来的测试应用，群主需要研读企业提供的测试需求文档，并且划分任务。任务划分完成之后也只在该群可见，这样做的好处是有时候老师想要组织一场考试，可以以班级为单位建立一个班级群，然后在群里创建任务让学生进行测试，老师也就是群主还可以给学生们的测试报告进行打分。创建任务的方式与管理员为任务广场创建任务的方式相同，都是点击创建按钮然后跳出一个弹出框。界面样式如图 4.23 所示使用模态框的考虑是出于模态框可以最小的入侵页面，并且是用户的注意力全部专注与模态框上的内容，而且还能丰富用户的使用体验。

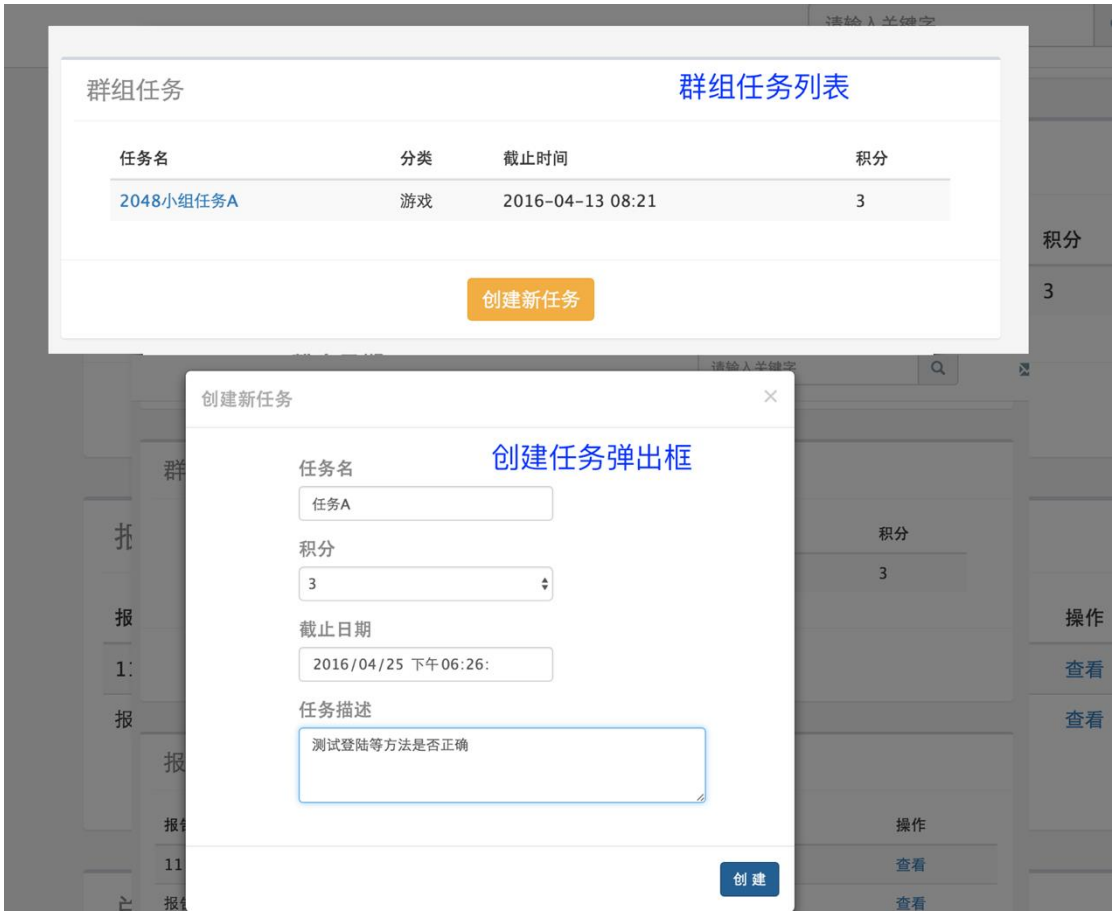

图 4.23 任务列表和创建弹出框

4.7 本章小结

第四章主要讲了 kikbug 前端应用的设计与实现，先按组件讲了程序中不同组件的功能和实现原理。然后按角色分别介绍不同用户角色的页面设计原则以及关键页面的实现方式和显示结果，重点介绍了其中的一些实现难点。

第五章 总结和展望

5.1 总结

kikbug 项目从年初设计实现开始，到四月初上线使用，期间我们进行了多次评估，做出了许多次的修改和改进。现在从投入使用的情况来看，kikbug 的各个组成部分移动应用端、网页端、后端服务都没有出现较大的问题。我们开发 kikbug 的初衷是为了将教学和实践能够结合，整合学校和企业的资源，各取所需互利发展，从现在的运营情况来看，这一目标的实现是完全有可能的。截止到本文撰写时间，已有好几个应用在 kikbug 平台上进行了测试，提交的测试报告数量也很可观。未来我们还将大规模的推广 kikbug 平台，努力使更多的学生和企业能够参与进来，为众测提供更多新思路新想法。

本文首先介绍了 kikbug 众测项目产生的背景以及实现 kikbug 平台的初衷和愿景。接着详细的阐述了 kikbug 平台的需求和整体架构设计，详细介绍了 kikbug 前后端分离设计的原因和实现的方法。最后本文详细的介绍了 kikbug 前端的详细设计和具体实现，并介绍了 angularJS 框架下各种组件的功能和实现方式。

另外，Kikbug 开发当中使用到了许多我们在企业实习时候学习到的知识和方法，比如：开发工具的选择、解决方案的查找、debug 的方法等等，这次项目的开发也让我们对这些知识的使用和理解更加深入了。

5.2 展望

从目前 kikbug 的运营结果来看，我们的初衷实现的非常不错，移动端和网页端配合有序，测试流程和结果也很好。但是目前平台上有一些工作还需要进行改进，首先一个是管理员和群主需要总结应用的细分报告并且给出一个总结报告，但是有时候细分报告的数量很大并且有内容重复，这就非常的浪费管理员和群主人员的时间，我们接下来的工作就是自动去重并且生成总结报告。另外一个是我们目前的报告反馈给出的是企业评分和群主评分，但是这个评分比较主观而且不一定能够及时反馈，所以接下来我们打算进行自动评分。总之，我们接下来的工作就是为 kikbug 加入更多自动化的特性，尽量减少用户的工作量。

参考文献

[1] AngularJS 官方网站，http://docs.angularjs.cn

[2] Gulp 中文官方网站，http://www.gulpjs.com.cn/

[3] Bower 官方网站，http://bower.io/

[4] 又拍云存储，https://www.upyun.com/zh/index.html

[5] Nginx 官网，http://nginx.org/

[6] “The power of crowds”，2016，Michelucci

[7] “A survey of the use of crowdsourcing in software engeer”，2015，Ke Mao, Licia Capra, Mark Harman and Yue Jia

[8] “Hypertext Transfer Protocol”[S],RFC 2616,June 1999

[9] “The JSON Data Interchange Format”[S],ECMA 404,October 2013

[10]“HTTP State Management Mechanism”[S],RFC 6265,April 2011

[11]Jesse James Garrett,“Ajax: A New Approach to Web Applications”[J], February 2005

[11] 骆斌,丁二玉,《需求工程—软件建模与分析》[M],2009

[12] 李宝敏. 动态网站设计与开发应用教程[M] . 清华大学出版社.

致谢

Kikbug 从需求的讨论到开发再到上线运行，经历了三个多月的时间，这期间我们遇到了一些困难但是都很好的克服了，在此，我要对给予我关心的人们表示诚挚的感谢。

首先要感谢的是我的指导老师陈振宇教授给与了我支持和帮助，从项目的开始阶段陈老师就对我们进行了认真的指导，帮助我们缕清需求、挖掘新思路。并且在我们项目的进行过程中认真的对我们的阶段成果进行评估，指出我们的不足之处并且为我们提出了许多很好的改进意见。他清晰的思路、开阔的见识和充沛的精力也让我们非常佩服，对我们的开发过程和未来的道路有着非常积极的影响。

其次我要感谢的是我的组员，由其是张玄同学，他的知识面和实践经验非常的丰富，在开发过程中他一直是我们组的技术骨干，为我们的项目提出了许多的新的技术和想法，在同他的合作中我学习到了许多新的知识和技能。

然后要感谢我的父母，因为他们对我的爱护和支持，我才能够做自己想做的事情，他们的鼓励也是我前进的动力。

最后要感谢周围所有帮助过我，支持鼓励过我的老师、同学、朋友和家人们。同时也向评阅本论文和答辩委员会的各位老师们致以诚挚的谢意,感谢你们的辛勤工作。