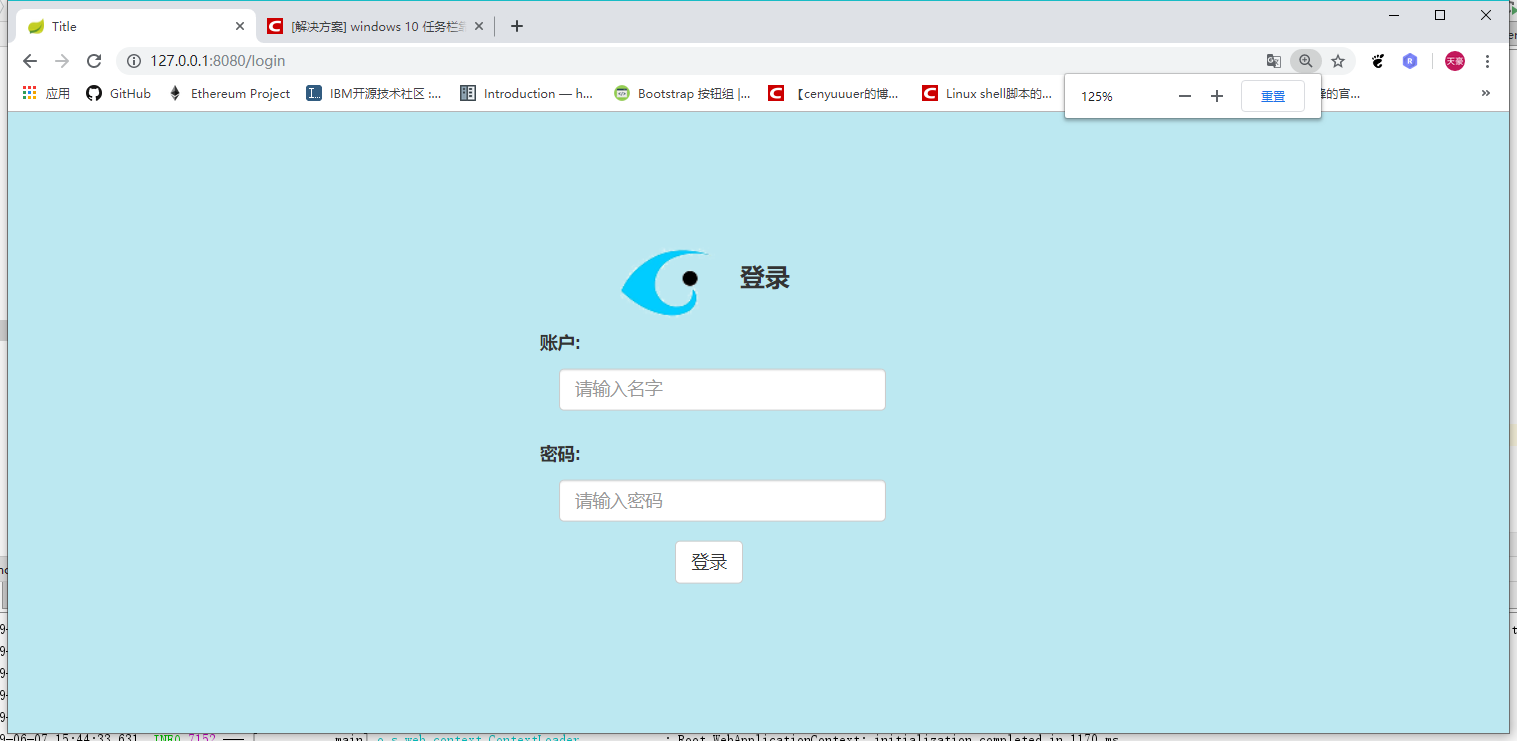
## 6.1登录及注册模块设计

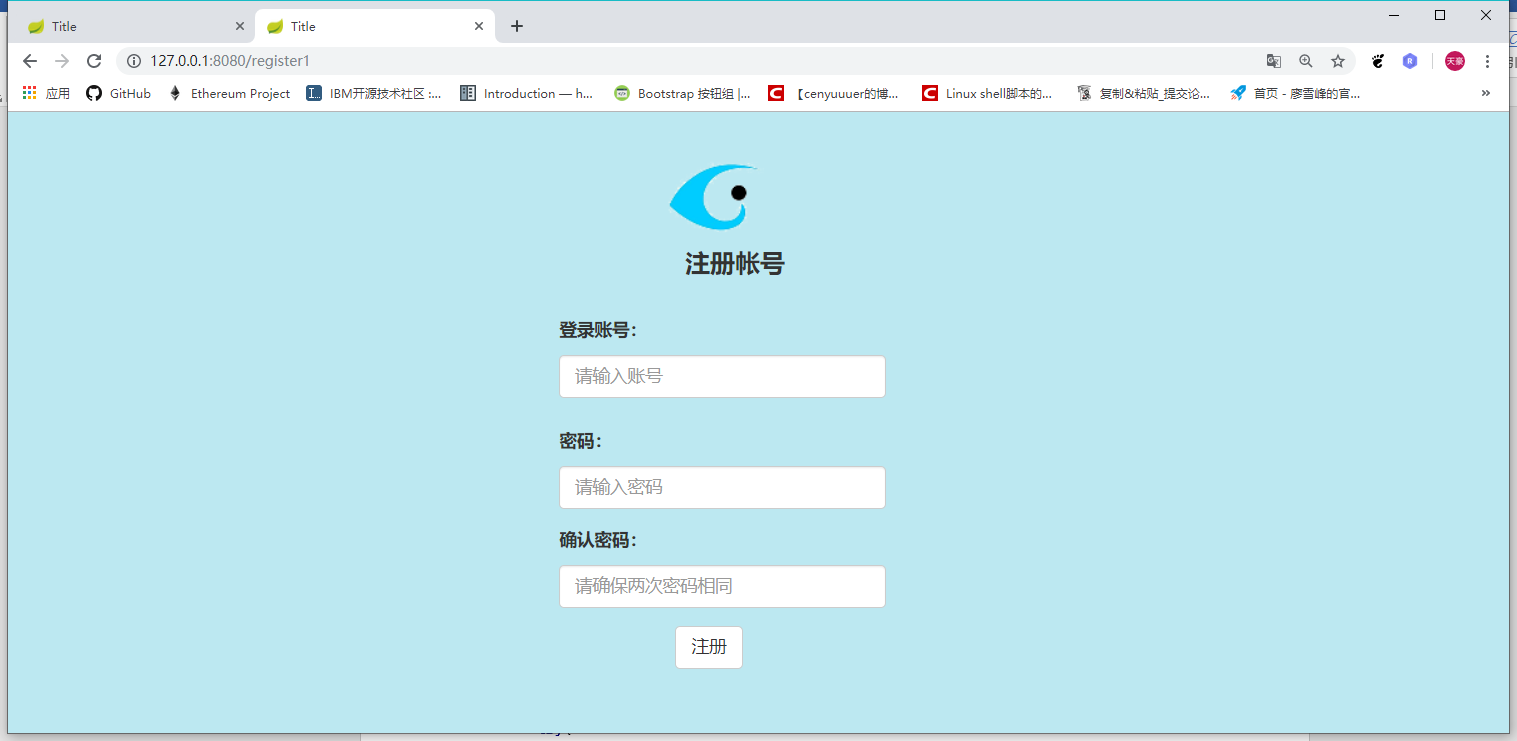
在登录模块设计中，用户根据用户名、密码登录系统。本系统分别为用户和管理员设置了不同的登录界面，利用html和css绘制界面的主题，组件库使用的是基于jquery的bootstrap，页面解释器使用的是thymeleaf。使用javascript对验证表单的元素进行基本的验证，不满足正确格式的用户名会给出提示，如果元素的的内容格式完全正确才会将数据提交至后台。处于用户信息安全的信息考虑，在注册登录过程中会使用shiro安全框架对用户的密码进行加密，加密方法使用md5加密方式，加salt迭代十次得到最终密码加密字符串，后台用此字符串与数据库中的字符串进行比对，若一样则登陆成功过，若不一样则赶回错误提示。登陆页面如图6.1所示。



主要程序代码：

**public int** checkPass(String id, String pwd) {  
 Md5Hash md5Hash = **new** Md5Hash(pwd,**"xth.com"**,10);  
 pwd = md5Hash.toString();  
 UsernamePasswordToken token = **new** UsernamePasswordToken(id,pwd);  
 DefaultSecurityManager securityManager = **new** DefaultSecurityManager();  
 AdminRealm adminRealm = **new** AdminRealm();  
 adminRealm.setPwd(**userMesMapper**.selectByPrimaryKey(id).getPassword());  
 securityManager.setRealm(adminRealm);  
 SecurityUtils.*setSecurityManager*(securityManager);  
 Subject subject = SecurityUtils.*getSubject*();  
 **try** {  
 subject.login(token);  
 **if** (subject.isAuthenticated()) {  
 **return** 1;  
 }  
 }**catch** (AuthenticationException e){  
 **return** 0;  
 }  
 **return** 0;  
}

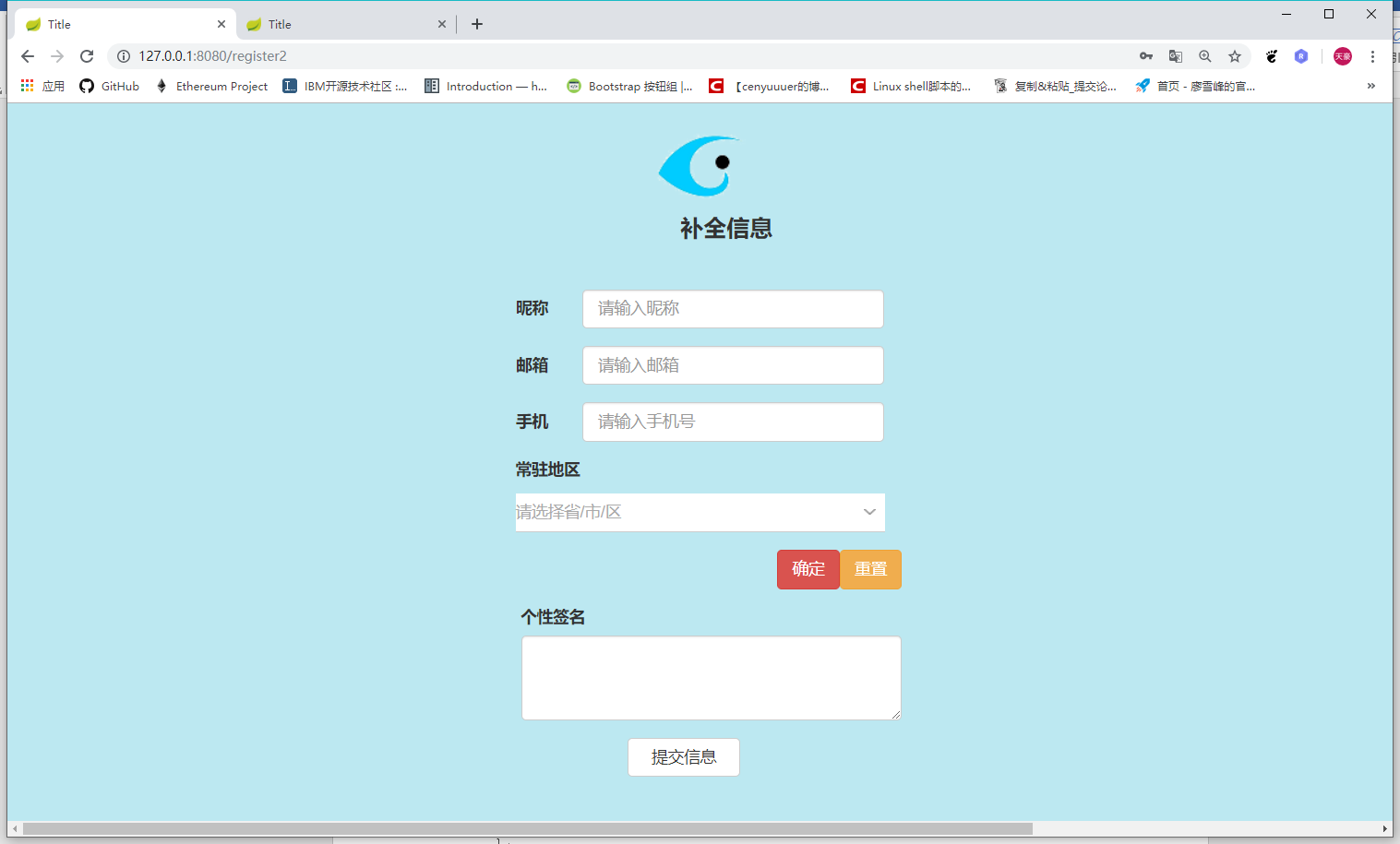
在注册模块中使用javascript对用户输入的两次密码进行一致性校验。如果两次密码输入的是相同的，用户注册所使用的账号密码信息会传至后端，后端会检验账号是否已经注册过，若已经注册过，则会返回账号已经注册过的消息提示。若前后端审核都通过，那么会跳转到输入个人基本信息的界面。注册账号的页面如图6.2所示。



注册逻辑代码：

**public int** insertUser(String userId,String password){  
  
 Md5Hash md5Hash = **new** Md5Hash(password,**"xth.com"**,10);  
 password = md5Hash.toString();  
 UserMes um = **new** UserMes();  
 um.setUserId(userId);  
 um.setPassword(password);  
 **try**{  
 **userMesMapper**.insert(um);  
 **levelScoreMapper**.insert(userId);  
 }**catch** (Exception e){  
 System.***out***.println(e.getMessage());  
 **return** 0;  
 }  
 **return** 1;  
}

在注册模块的输入个人信息模块中，需要用户输入昵称、邮箱、手机、常驻地区、以及个性签名。其中常驻地区用于在用户访问悬赏大厅的时候职能的给用户调转到其常驻地区的悬赏信息页面。在本页面中所有信息都必须填写，使用javascript进行信息格式的校验，若数据无误则传入后端。注册个人信息页面如图6.3所示。

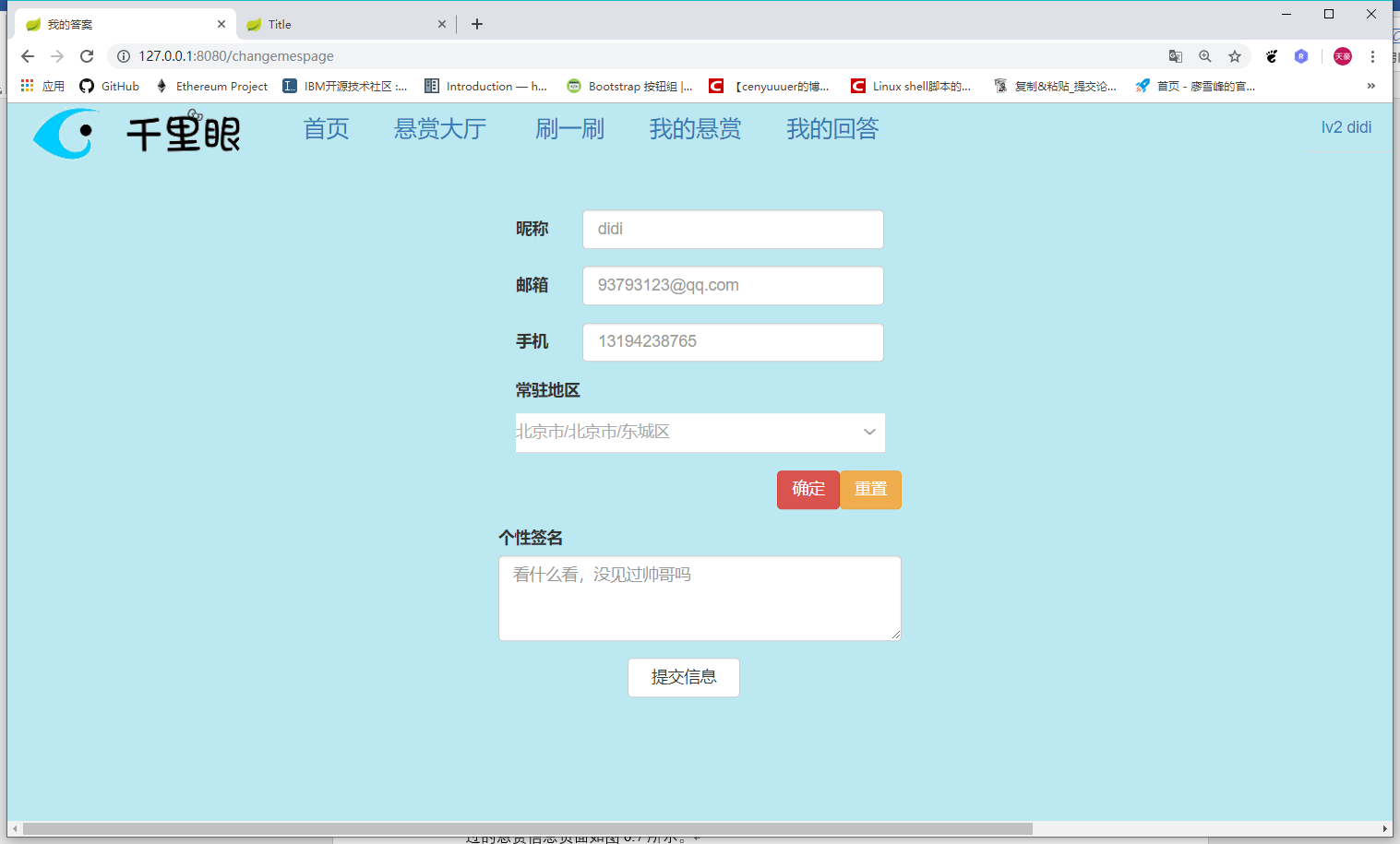


主要代码：

**public** String register2(HttpServletRequest request,HttpServletResponse response) **throws** IOException {  
 HttpSession session = request.getSession();  
 String userId = (String) session.getAttribute(**"userId"**);  
 String nickName = (String)request.getParameter(**"nickName"**);  
 String email = (String)request.getParameter(**"email"**);  
 String phone = (String ) request.getParameter(**"phone"**);  
 String region = (String) request.getParameter(**"city-picker3"**);  
 String signature = request.getParameter(**"signature"**);  
 session.setAttribute(**"nickName"**,nickName);  
 UserMes um = **new** UserMes();  
 um.setUserId(userId);  
 um.setNickname(nickName);  
 um.setEmail(email);  
 um.setPhone(phone);  
 um.setRegion(region);  
 um.setSignature(signature);  
  
 **if**(**userServiceimpl**.updateMes(um)==1)  
 **return "index"**;  
 **else** {  
 response.setContentType(**"text/html;charset=utf-8"**);  
 PrintWriter out = response.getWriter();  
 out.println (**"<script language=javascript>alert('注册失败，请稍后再试')</script>"**);  
 }  
 **return "/"**;  
}

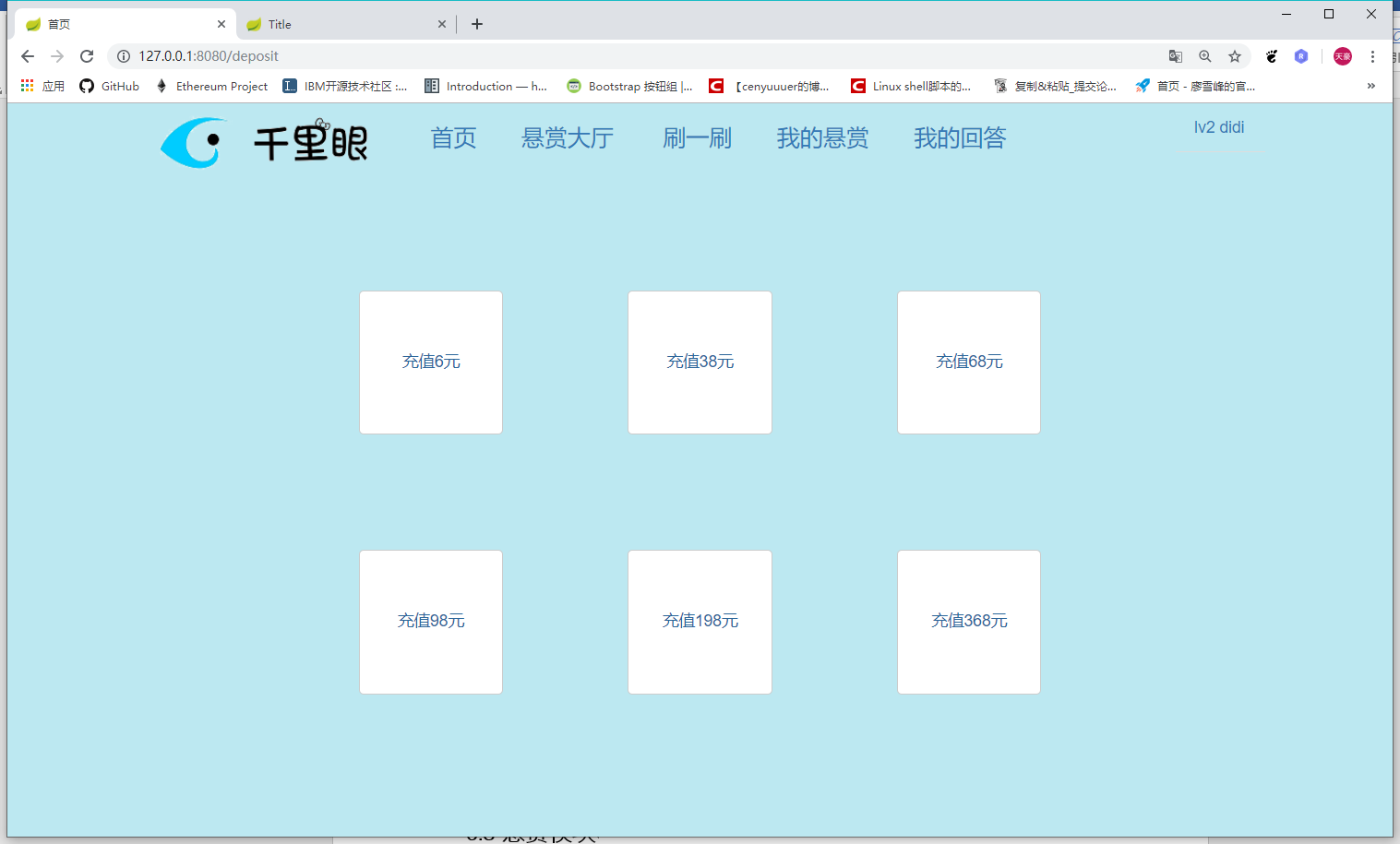
## 6．2个人信息维护模块

在个人信息维护模块中，用户可以维护修改自己的个人信息，包括密码、昵称、邮箱、手机号、个性签名等信息。为保证用户仅能修改其自身的账户信息及密码，后台会审核当前用户是否具有修改目标用户信息及密码的权限，若具有权限则允许修改，若没有权限则返回警告提示。修改个人信息模块如图6.4所示。



密码修改界面会要求用户输入两次密码，确保用户输入密码的正确性，因为在后台会审核当前用户是否具有修改目标用户的密码权限，所以这里并没有要求用户在输入一次密码，简便了操作，修改密码界面如图6.5所示。

此系统模拟了一个充值页面，用户可以在此页面点击按钮进行充值。充值页面如图6.6所示。



## 6.3悬赏模块

悬赏模块其中包括展示已经发布过的悬赏大厅、个人悬赏信息、展示此悬赏的答案、发布图片悬赏和发布视频悬赏四个子功能。

展示发布过的悬赏信息，包括具体地点、所在区域、当前悬赏状态、悬赏奖金、发布时间以及可选择查看答案以及删除悬赏的操作。悬赏状态包括已解决和未解决。查看答案会弹出回答该悬赏答案的页面，若发布该悬赏的用户已经在所有回答中选择了中标答案，则此页面将仅作于展示悬赏答案使用，若并未选择中标答案，则可在此页面选择一个答案为最终中标答案。删除悬赏操作会以弹窗的方式让用户再次确认删除该悬赏，以防止误删。展示发布过的悬赏信息页面如图6.7所示。



在展示个人悬赏信息页面，若有其他用户回答了此悬赏，发布悬赏的用户可以在对应悬赏的条目下看到一个查看回答按钮，点击查看回答可以跳转到回答界面。若用户并未为此悬赏选择最优答案，则在每个答案提交人昵称旁边会生成一个中标按钮，用户可以选择一个最优答案。若已经选择了最优答案，则在最优答案提交人的昵称旁生成一个中标的标签。同时针对多人回复一个悬赏的情况，本论坛在前端展示进行了分页处理，每个用户的回答自成一页。查看答案页面如图6.8所示



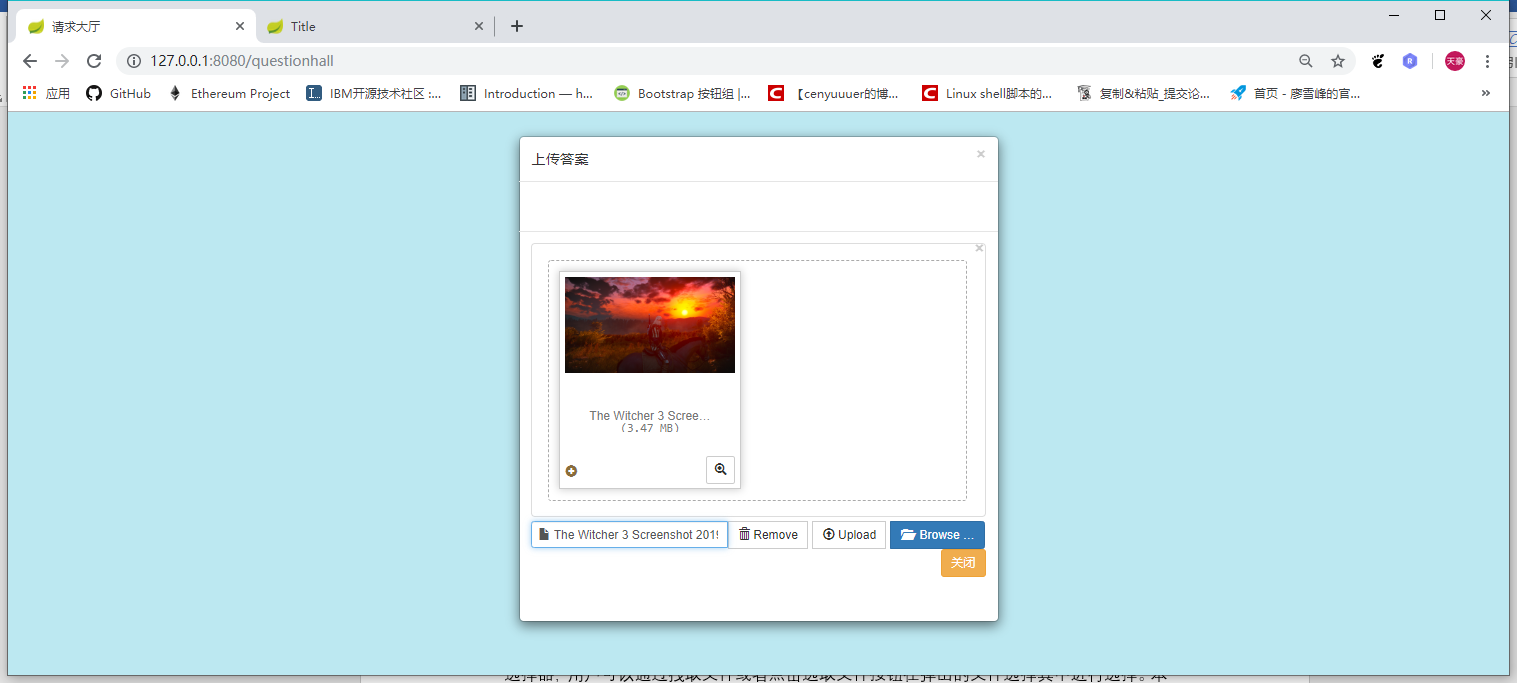
悬赏大厅，用户可以在此页面中浏览某一地区的未解决的悬赏，展示的条目默认根据选上发布的时间进行排序。条目展示内容包括具体地点、所在区域、奖金数额、发布时间和操作。悬赏大厅页面如图6.9所示。



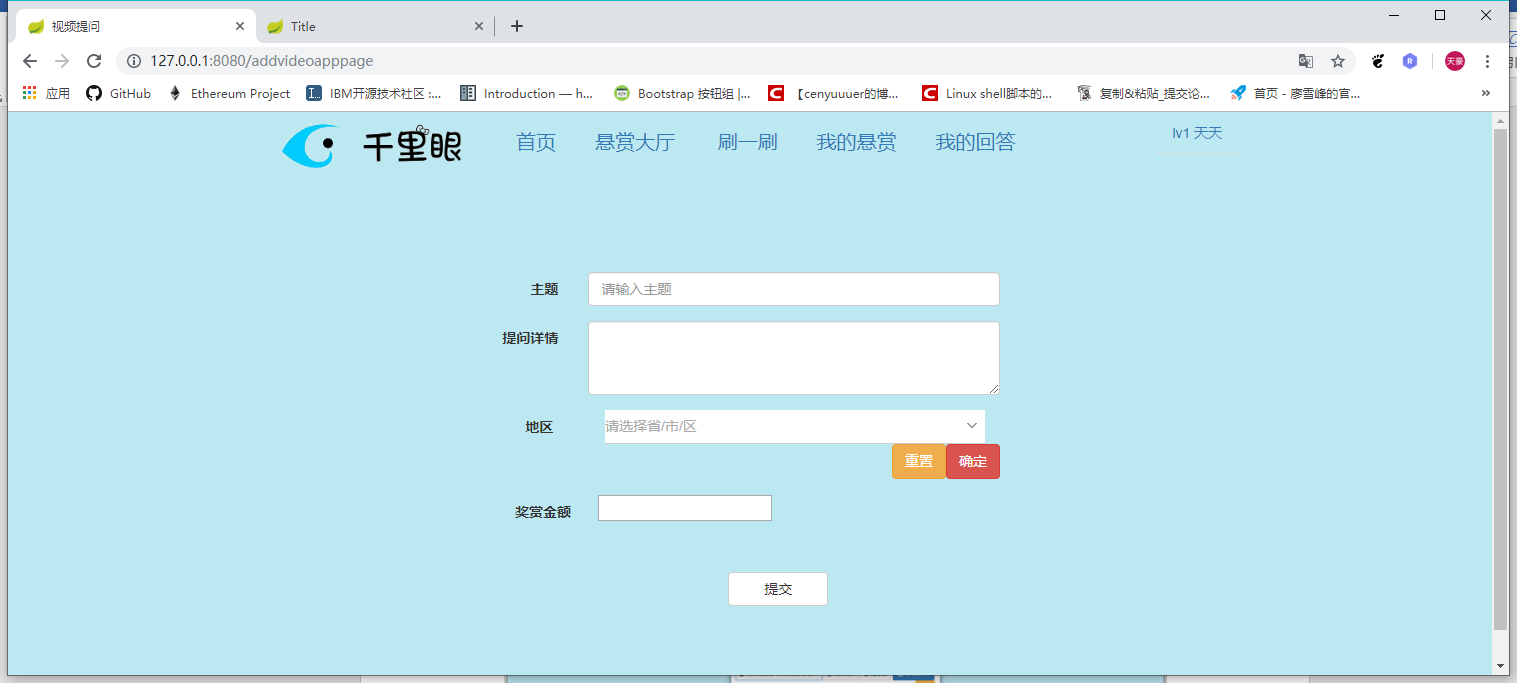
悬赏大厅中每条悬赏条目包括查看介绍以及回答操作。点击查看介绍会弹出一个bootstrap组件model模态框，在此模态框中展示发布用户填写的具体要求。查看介绍模态框如图6.10所示。



在悬赏大厅中点击回答按钮，同样会弹出一个model模态框。在此模态框中设置了文件 选择器，用户可以通过拽取文件或者点击选取文件按钮在弹出的文件选择其中进行选择。本模块中所使用的文件选择器设置为最多上传十张照片或者一个视频，文件选择器支持预览功能，用户可以点击预览按钮以查看大图或者播放视频。回答模态框如图6.11所示。



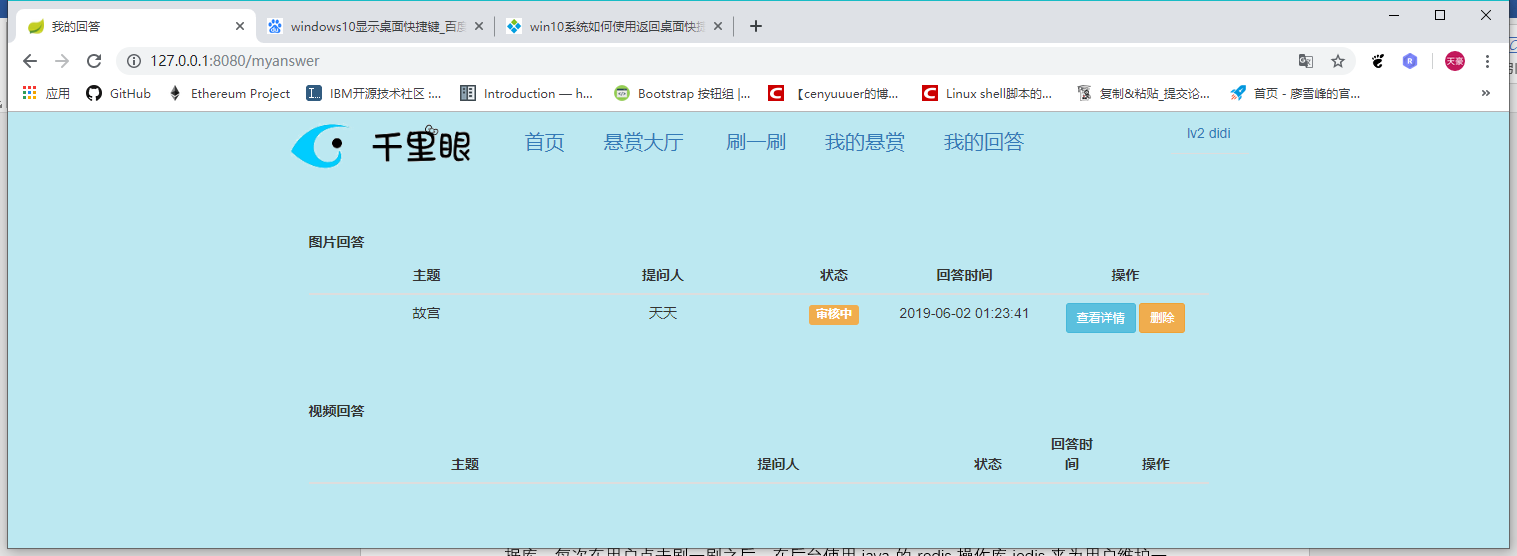
在我的悬赏页面可以发布新的悬赏。悬赏种类包括图片悬赏和视频悬赏。悬赏信息包括悬赏具体地点、悬赏地点所在地区、悬赏详细介绍、悬赏金额。添加悬赏信息页面如图6.12所示。



## 6.4答案模块

答案模块其中包括个人答案信息、最佳答案排行两个子页面。

个人答案信息页面，用户可以在此页面看到自己回复悬赏的答案信息，其中包括悬赏地点、悬赏地点所在区域、答案状态、回答时间、以及查看详情和删除亮相操作。答案状态包括审核中、已审核、中标和违规。只用已审核和中标的答案会对其他用户进行展示。状态码标签根据状态的不同会呈现不同的颜色。点击删除链接，系统会要求用户重复确认以防止用户的误操作。点击查看详情链接可以查看用户自己回答的具体内容，同时也可以看到一个悬赏下其他用户提交的答案，方便对自身的答案进行修改。回答个人信息页面如图所示。



## 6.5刷一刷模块

刷一刷模块用于随机为用户推送优秀的图片悬赏的答案和视频悬赏的答案。使用户能够在查看高质量的答案以及不断优化自己的答案内容。本模块采用的解决方案是使用redis数据库，每次在用户点击刷一刷之后，在后台使用java的redis操作库jedis来为用户维护一个属于该用户的刷一刷队列，刷到一个图片悬赏的答案或者视频悬赏的答案就向redis对应的key中追加一个答案id。队列有效时间设置为10分钟，每次用户有新的刷一刷操作都会更新队列的有效时间。为了保证给用户推送的图片悬赏答案和视频悬赏答案具有一定的随机性，存储数据库的图片悬赏答案和视频答案所用的数据结构使用的是HashSet。刷一刷页面如图6.13所示。

主要代码：

@RequestMapping(value = **"shuayishu"**)  
**public** String shuayishua(HttpServletRequest request, ModelMap modelMap, HttpServletResponse response) **throws** IOException {  
 String userId = (String) request.getSession().getAttribute(**"userId"**);  
 modelMap.addAttribute(**"level"**,level);  
 SaPicAnswer saPicAnswer = **saService**.getRandomPic(userId);   
 **if**(saPicAnswer==**null**){**return "nilshua"**;  
 }  
 **else** {  
 modelMap.addAttribute(**"saPicAnswer"**,saPicAnswer);  
 }  
 **return "picshua"**;  
}

## 6.6等级模块

等级模块提供了计算等级的功能。本论坛的等级经验值由两部分构成，分别是在线时间和回答悬赏奖励的回答分数。在线时间的统计的实现方法是在Mysql数据库中维护一个当次在线时间的统计表、等级划分经验值表、总在线时间和答案分数的汇总表，用户初次登录的时候会在表中插入数据，数据项包括本次用户id、本次登录时间、最后一次操作时间。用户每次执行操作的时候都会更新对应的最后一次操作时间。统计回答分数，用户提交的图片悬赏答案或者视频悬赏答案如果通过审核，则会获得30点回答的经验值，若答案被发布悬赏的用户选中为最优答案，则会获得200点经验值，在线时长和答题经验值有着不同的权重。每次跳转一个新页面之后后端都会重新计算当前用户的最新等级。

等级计算公式：

计算等级的主要函数如下：

**public int** getLevel(String userId){  
 LevelScore levelScore = **levelScoreMapper**.selectSumScoreByUserId(userId);  
 **int** sumScore = (**int**) (levelScore.getAnswerScore() +0.1\*levelScore.getLoginTime());  
 **int** level = **levelMapper**.selectLevel(sumScore);  
 **return** level;  
}

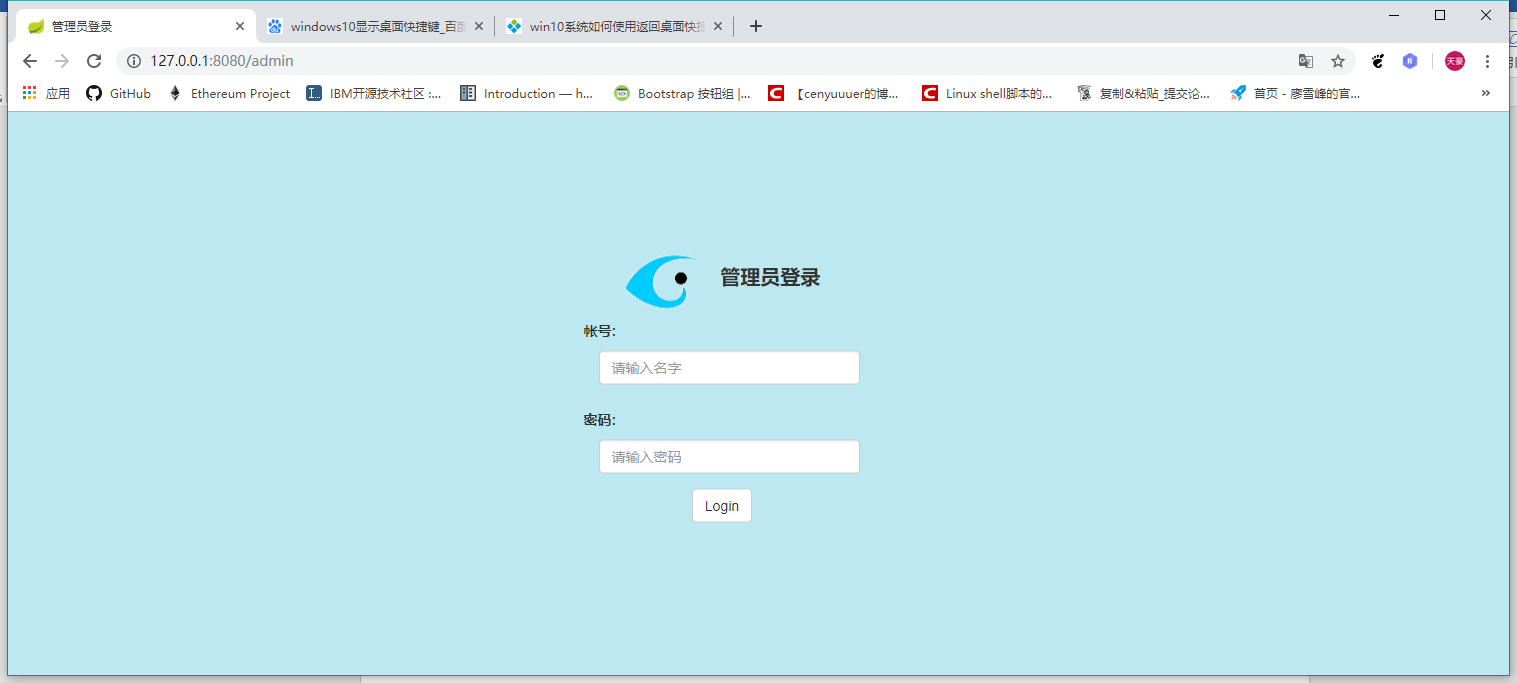
为了对用户登录时间进行及时的审核，在main方法启动前，在Springboot所有Bean注入完成后新建一个线程，该线程用于统计所有用户在当此登录的有效时长。统计当此时长代码如下：

**public void** run(){  
 **while** (**true**){  
 List<TimeSatistic>tss = **timeSatisticMapper**.selectAll();  
 **if**(tss.size()==0){*//没有账户登录* **try** {  
 *sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }**else** {*//有账户在登录* **for** (TimeSatistic ts: tss  
 ) {  
 **if**(((**new** Date().getTime()-ts.getLatestoptime().getTime()))>1000\*60\*10){  
 *//超过时间,记录时间* **timeSatisticMapper**.deleteByPrimaryKey(ts.getId());  
 String userId = ts.getUserId();  
 LevelScore levelScore = **levelScoreMapper**.selectSumScoreByUserId(userId);  
 levelScore.setLoginTime((**int**) (levelScore.getLoginTime()+(ts.getLatestoptime().getTime()-ts.getLogintime().getTime())/1000/60));  
 **levelScoreMapper**.updateByPrimaryKey(levelScore);  
 }  
 }  
 **try** {  
 *sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

## 6.7管理员审核模块

管理员审核模块包括管理员的登录页面、审核图片悬赏答案、审核视频悬赏答案的功能。

管理员登录模块类似于用户登录模块，管理员在前端页面输入账号和密码，后端AdminController接收到数据传给下层AdminService进行密码加密以及账号密码信息的验证。管理员登录页面如图所示。



在管理员成功登录后，会跳转至图片悬赏答案审核页面，在该页面管理员可以对用户上传的图片悬赏答案进行审核以确保用户的答案中不会出现不良的信息。此管理员模块不作为后台信息管理使用，在此不显示用户的个人信息。本页面提供给了管理员两个处理链接，一个是审核通过链接，一个是答案违规链接。点击审核通过链接后端AnswerController会调用AnswerService进行答案状态信息的更改，若更改答案状态信息发生错误，也会想前台发出一个修改错误的提示。点击违规链接，后端修改该答案的状态信息为违规。其中悬赏答案只有在审核通过的情况下才会用户登录主页参与最高点赞排行榜以及刷一刷网页。审核页面如图所示。

## 6.8通知模块（待定）