学校代码：10128

内蒙古工業大學

面向对象综合设计小组报告

题　目　：英文单词游戏设计与实现

个人信息已删除

2019 年 1 月 18 日

目录

1 需求分析

1.1 用户管理

登陆用户，修改密码，注册用户，用户排名，用户积分，退出登陆。

1.2 积分管理

积分累计，积分排名，积分记录。

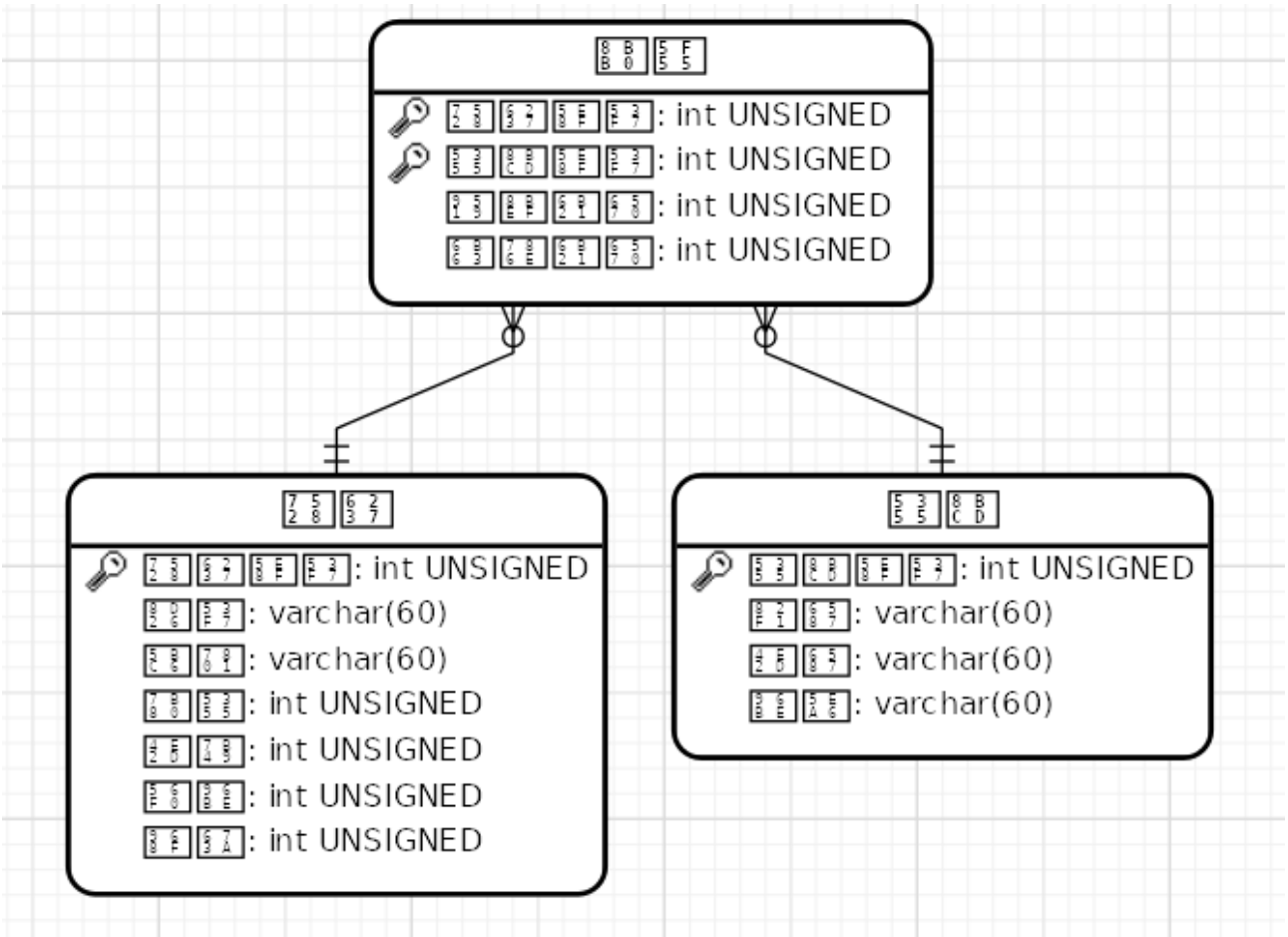
1.3 单词管理

单词的 CRUD，随机显示，单词和用户关联记录。单词分级，简单，中，难等级，选择单词的游戏。

1.4 主页设计

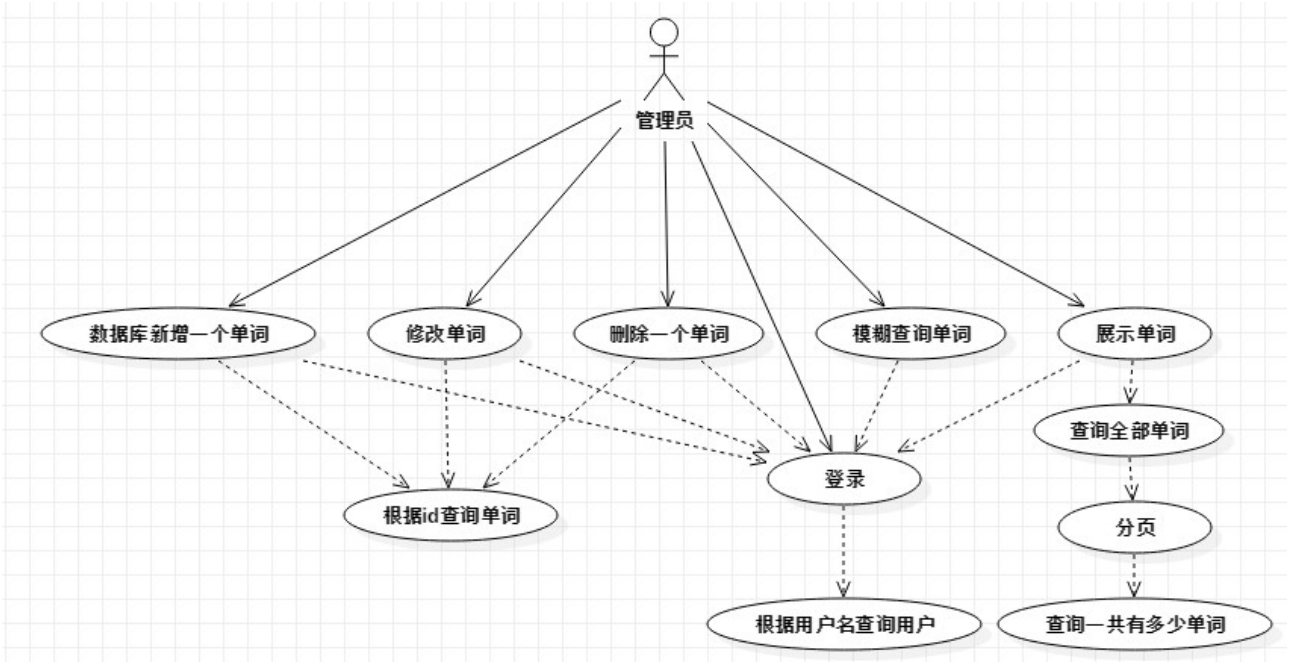
用户登陆成功之后，进入主页面，使用特效、动画等方式弹出单词，在 5 秒内，选择单词对应的中文词语，提交答案。选择正确，积分记录，选择错误，提示警告。进入下一轮。实时记录得分。

2 数据库设计

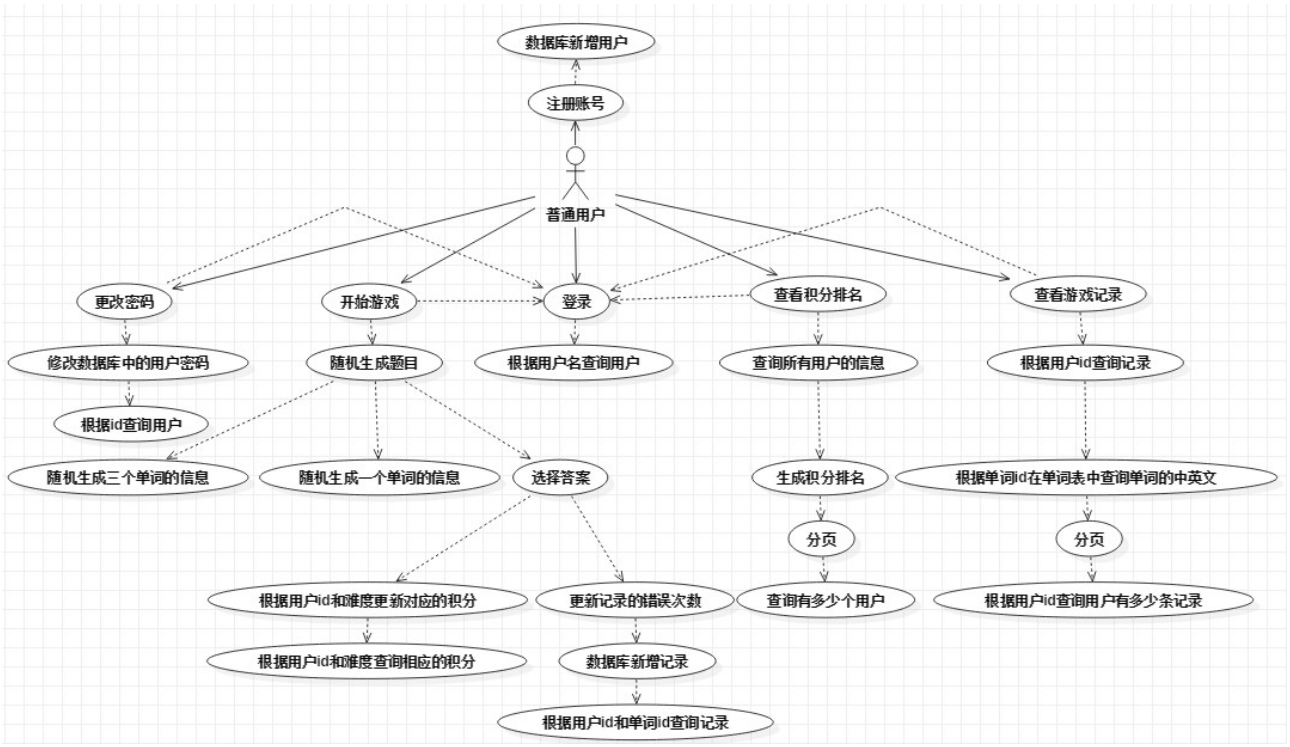


3 系统功能设计

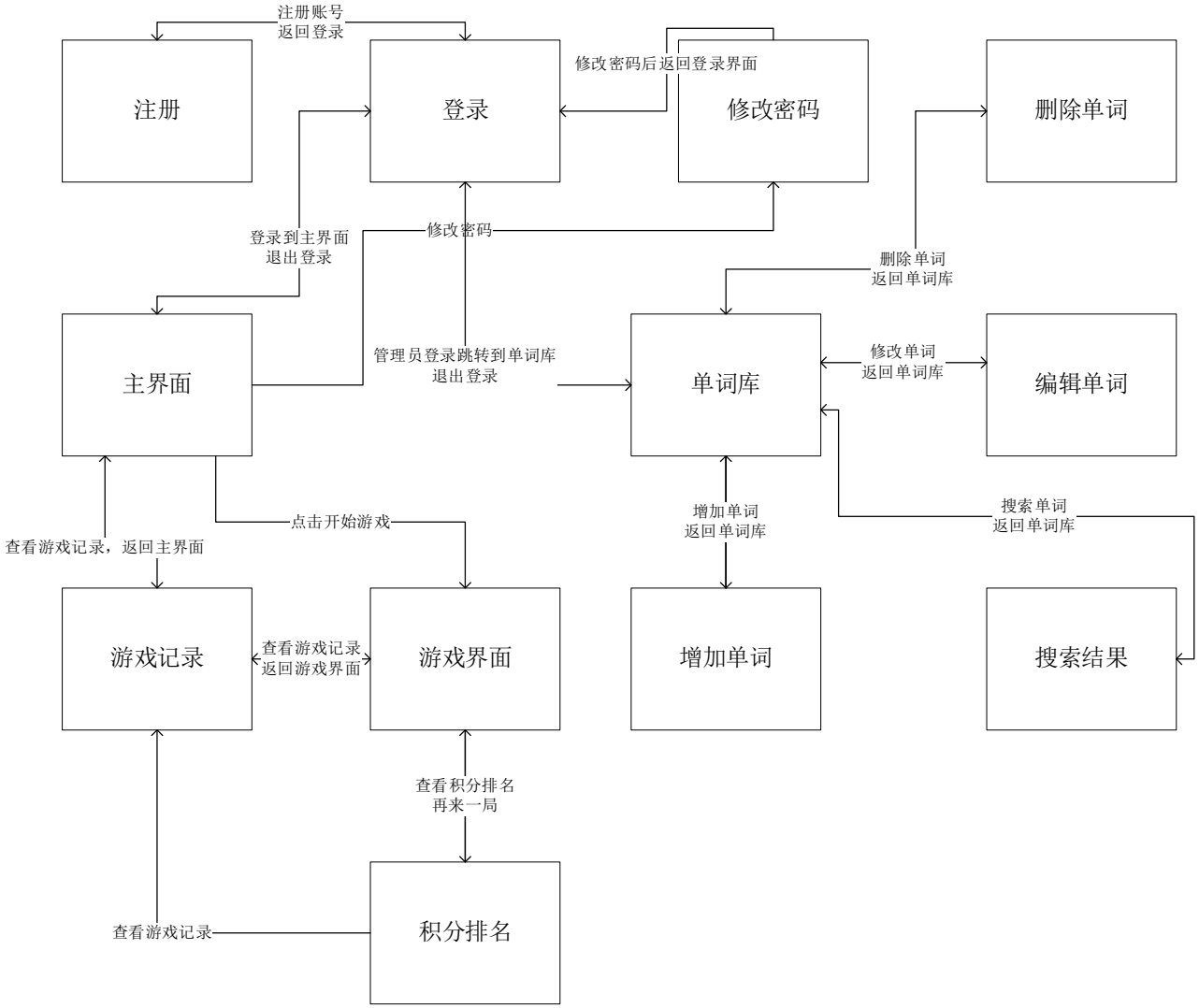
3.1 管理员用例图



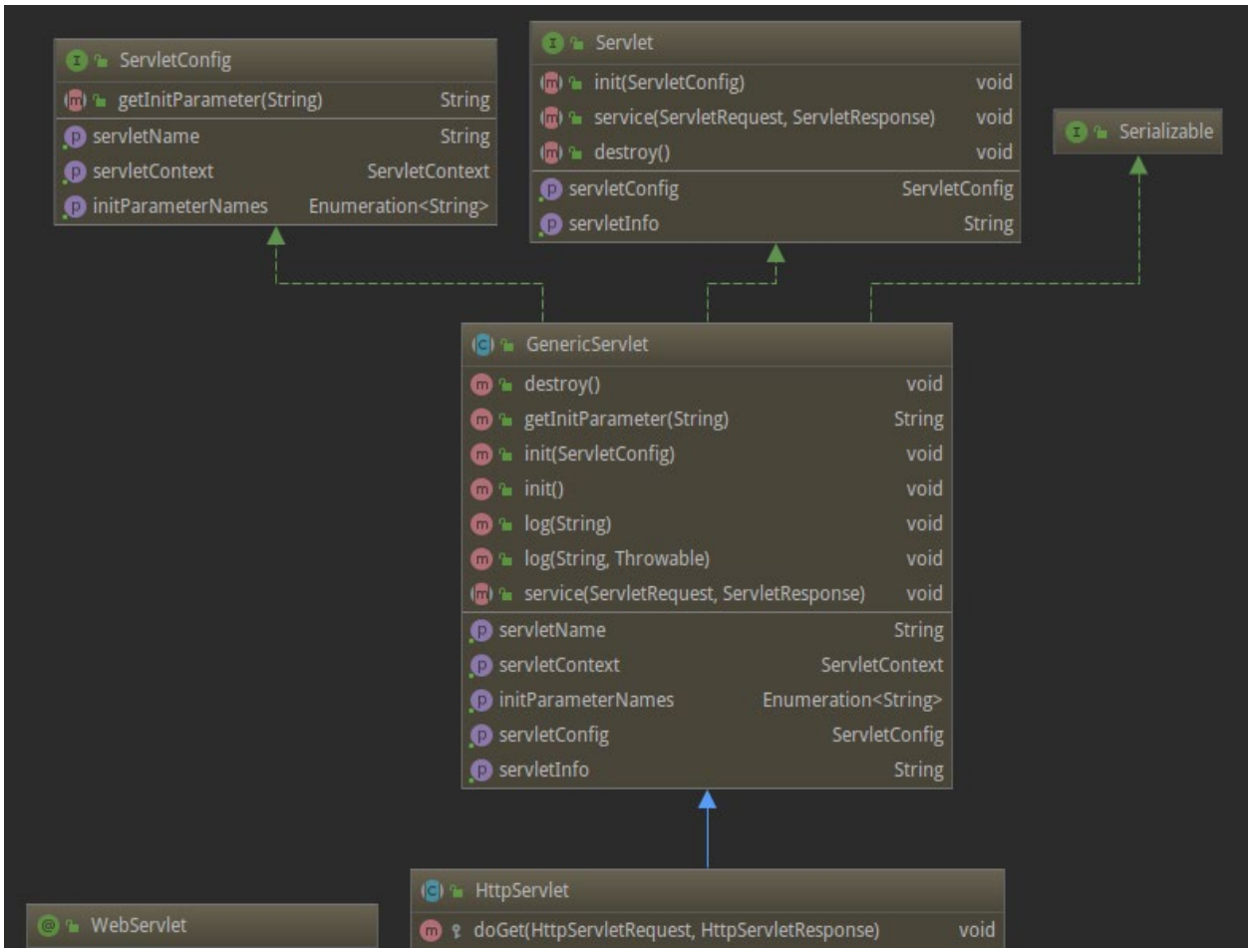
3.2 普通用户用例图

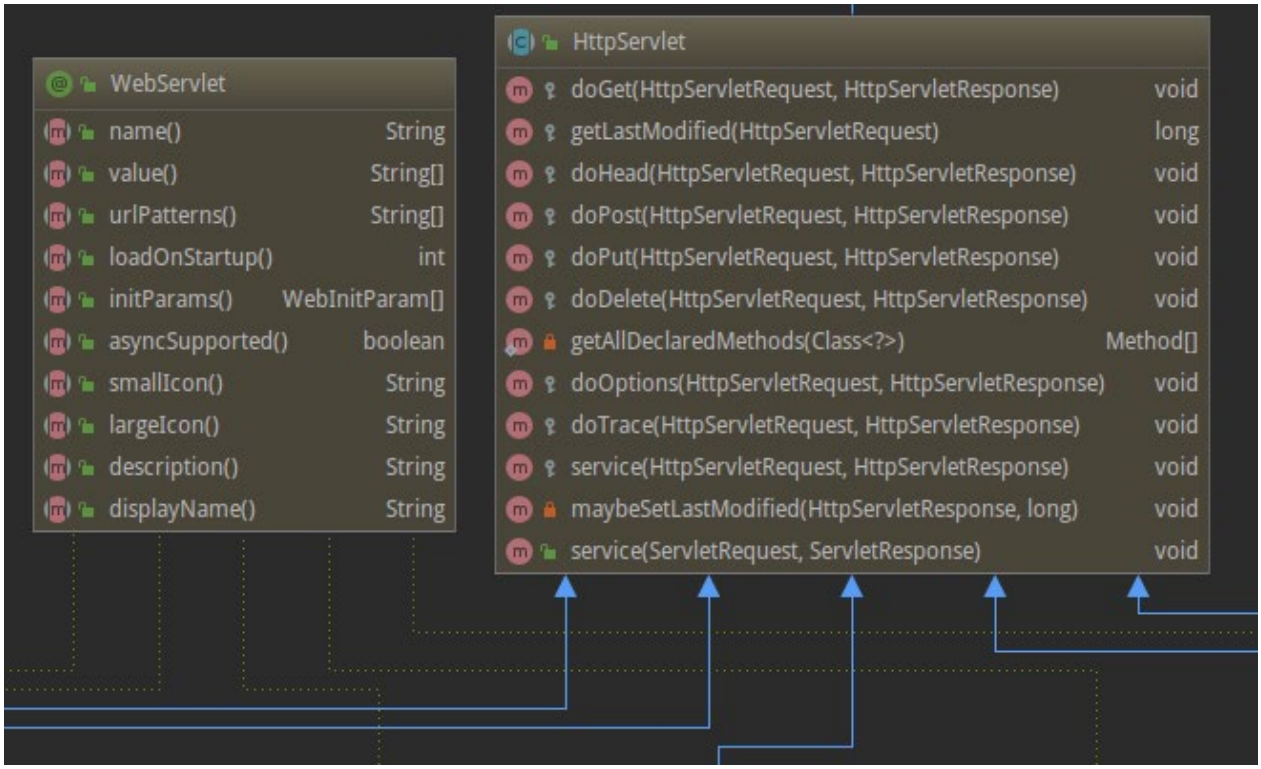


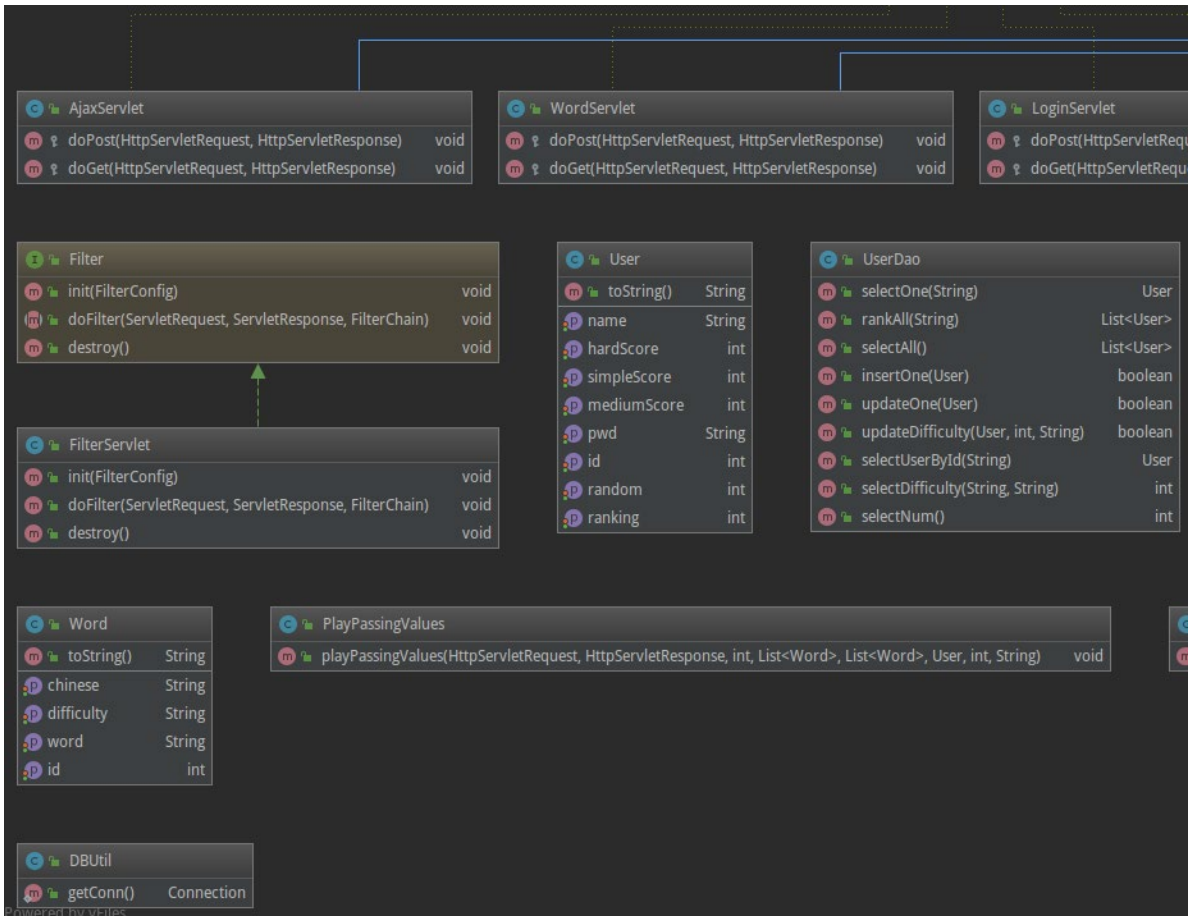
3.3 页面走向图

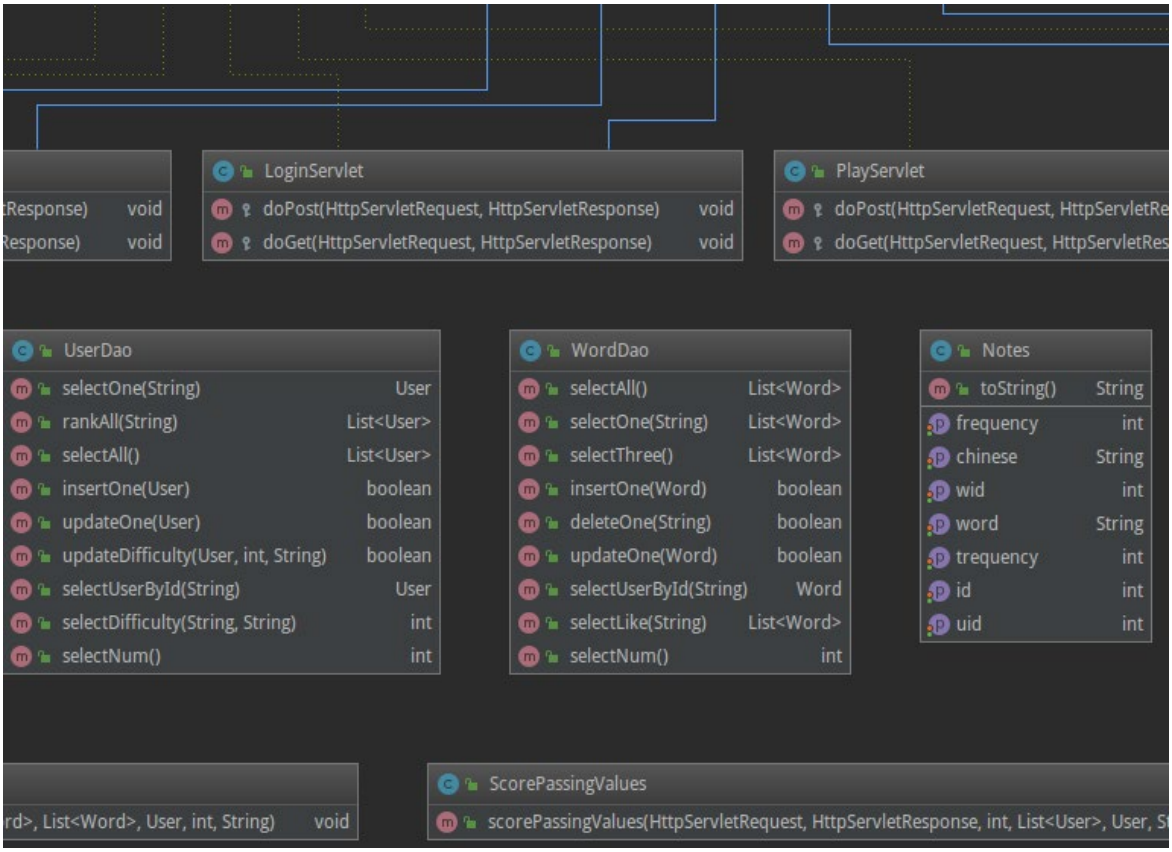

注册账号
返回登录
注册
登录
修改密码后返回登录界面
修改密码
删除单词
登录到主界面
退出登录
修改密码
删除单词
返回单词库
主界面
管理员登录跳转到单词库
退出登录
单词库
修改单词
返回单词库
编辑单词
增加单词
返回单词库
搜索单词
返回单词库
查看游戏记录，返回主界面
点击开始游戏
游戏记录
查看游戏记录
返回游戏界面
游戏界面
增加单词
搜索结果
查看积分排名
再来一局
查看游戏记录
积分排名

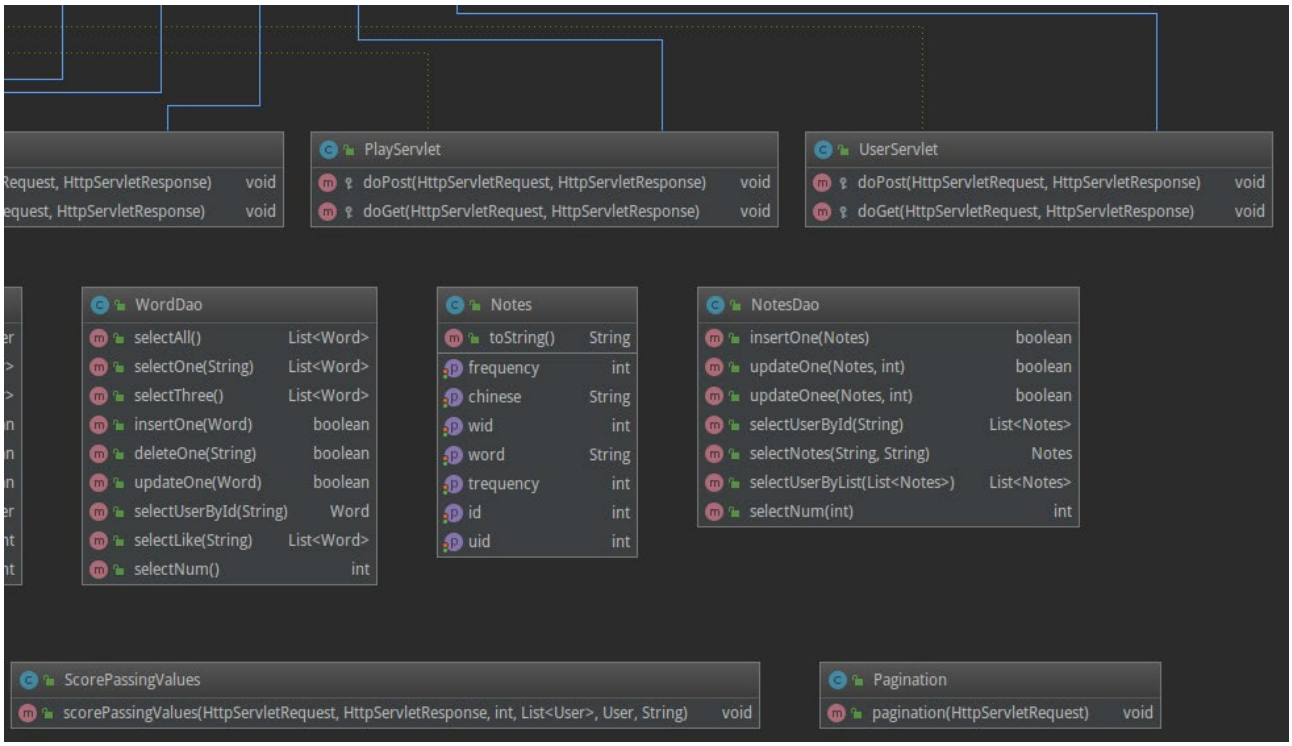
3.4 类图


ServletConfig
getInitParameter(String) String
servletName String
servletContext ServletContext
initParameterNames Enumeration<String>
Servlet
init(ServletConfig) void
service(ServletRequest, ServletResponse) void
destroy() void
servletConfig ServletConfig
servletInfo String
Serializable
GenericServlet
destroy() void
getInitParameter(String) String
init(ServletConfig) void
init() void
log(String) void
log(String, Throwable) void
service(ServletRequest, ServletResponse) void
servletName String
servletContext ServletContext
initParameterNames Enumeration<String>
servletConfig ServletConfig
servletInfo String
HttpServlet
doGet(HttpServletRequest, HttpServletResponse) void
WebServlet


WebServlet
name() String
value() String[]
urlPatterns() String[]
loadOnStartup() int
initParams() WebInitParam[]
asyncSupported() boolean
smallIcon() String
largeIcon() String
description() String
displayName() String
HttpServlet
doGet(HttpServletRequest, HttpServletResponse) void
getLastModified(HttpServletRequest) long
doHead(HttpServletRequest, HttpServletResponse) void
doPost(HttpServletRequest, HttpServletResponse) void
doPut(HttpServletRequest, HttpServletResponse) void
doDelete(HttpServletRequest, HttpServletResponse) void
getAllDeclaredMethods(Class<?>) Method[]
doOptions(HttpServletRequest, HttpServletResponse) void
doTrace(HttpServletRequest, HttpServletResponse) void
service(HttpServletRequest, HttpServletResponse) void
maybeSetLastModified(HttpServletResponse, long) void
service(ServletRequest, ServletResponse) void


AjaxServlet
doPost(HttpServletRequest, HttpServletResponse) void
doGet(HttpServletRequest, HttpServletResponse) void
WordServlet
doPost(HttpServletRequest, HttpServletResponse) void
doGet(HttpServletRequest, HttpServletResponse) void
LoginServlet
doPost(HttpServletRequ
doGet(HttpServletRequ
Filter
init(FilterConfig) void
doFilter(ServletRequest, ServletResponse, FilterChain) void
destroy() void
FilterServlet
init(FilterConfig) void
doFilter(ServletRequest, ServletResponse, FilterChain) void
destroy() void
User
toString() String
name String
hardScore int
simpleScore int
mediumScore int
pwd String
id int
random int
ranking int
UserDao
selectOne(String) User
rankAll(String) List<User>
selectAll() List<User>
insertOne(User) boolean
updateOne(User) boolean
updateDifficulty(User, int, String) boolean
selectUserById(String) User
selectDifficulty(String, String) int
selectNum() int
Word
toString() String
chinese String
difficulty String
word String
id int
PlayPassingValues
playPassingValues(HttpServletRequest, HttpServletResponse, int, List<Word>, List<Word>, User, int, String) void
DBUtil
getConn() Connection


LoginServlet
doPost(HttpServletRequest, HttpServletResponse) void
doGet(HttpServletRequest, HttpServletResponse) void
PlayServlet
doPost(HttpServletRequest, HttpServletRe
doGet(HttpServletRequest, HttpServletRes
UserDao
selectOne(String) User
rankAll(String) List<User>
selectAll() List<User>
insertOne(User) boolean
updateOne(User) boolean
updateDifficulty(User, int, String) boolean
selectUserById(String) User
selectDifficulty(String, String) int
selectNum() int
WordDao
selectAll() List<Word>
selectOne(String) List<Word>
selectThree() List<Word>
insertOne(Word) boolean
deleteOne(String) boolean
updateOne(Word) boolean
selectUserById(String) Word
selectLike(String) List<Word>
selectNum() int
Notes
toString() String
frequency int
chinese String
wid int
word String
trequency int
id int
uid int
ScorePassingValues
scorePassingValues(HttpServletRequest, HttpServletResponse, int, List<User>, User, S



4 系统实现

这是登录界面，下拉展示关于游戏的介绍，点击登录自动下拉到登录的地方。右上角的菜单打开后可以快速跳转到本页面的首页、介绍、登录三部分。

这是登录的界面。通过过滤器阻止未登录用户访问其他界面。若使用有管理员权限的账号登录，则登陆后跳转到单词库页面。

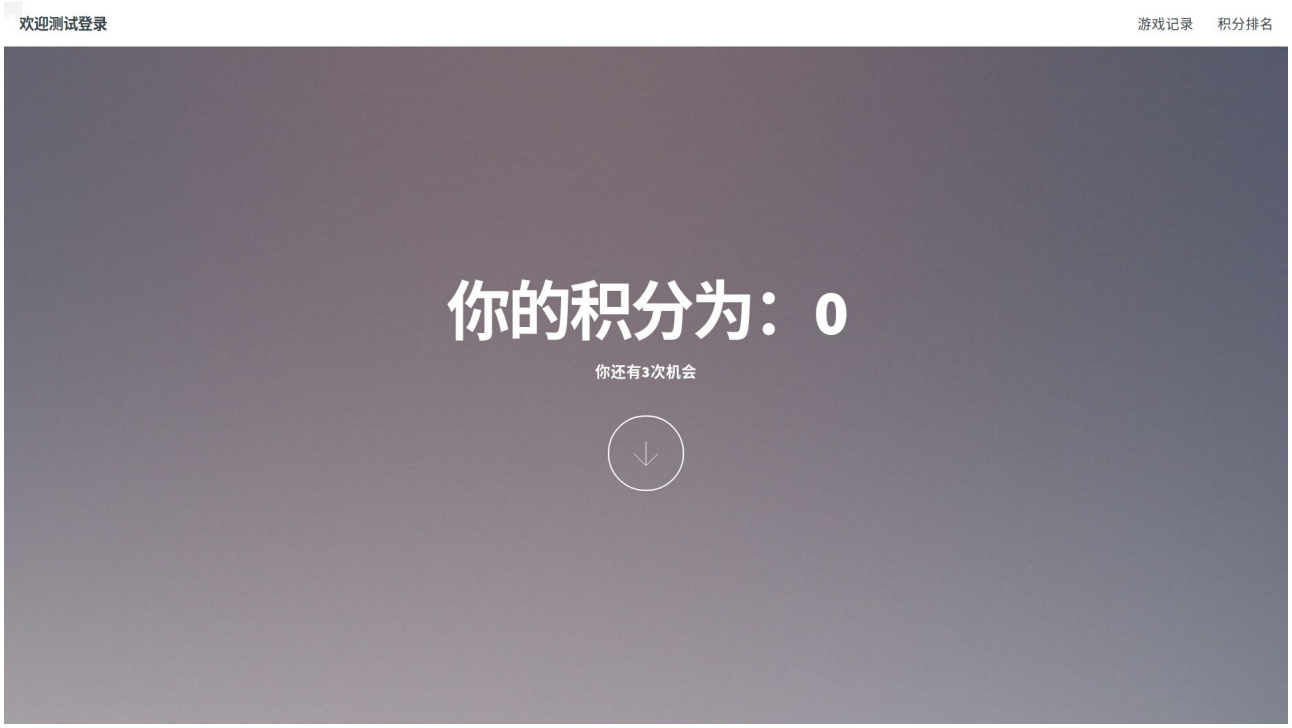
点击注册则跳转到注册页面，该页面用 Ajax 判断用户名是否已经被注册过。只允许注册普通权限的用户。具有管理员权限的账号需要在数据库中直接添加。

通过 Cookie 实现记住账号和密码。

登陆后进入游戏主界面，可以在这里点击修改密码，查看历史游戏记录，退出登录。还可以选择游戏的难度。简单难度为小学单词，中等难度为初高中单词，困难难度为大学以上的单词，随机难度会随机出现简单中等困难三个难度的单词。每一局游戏会有三次允许错误的机会。如果错误次数超过三次，则游戏结束。

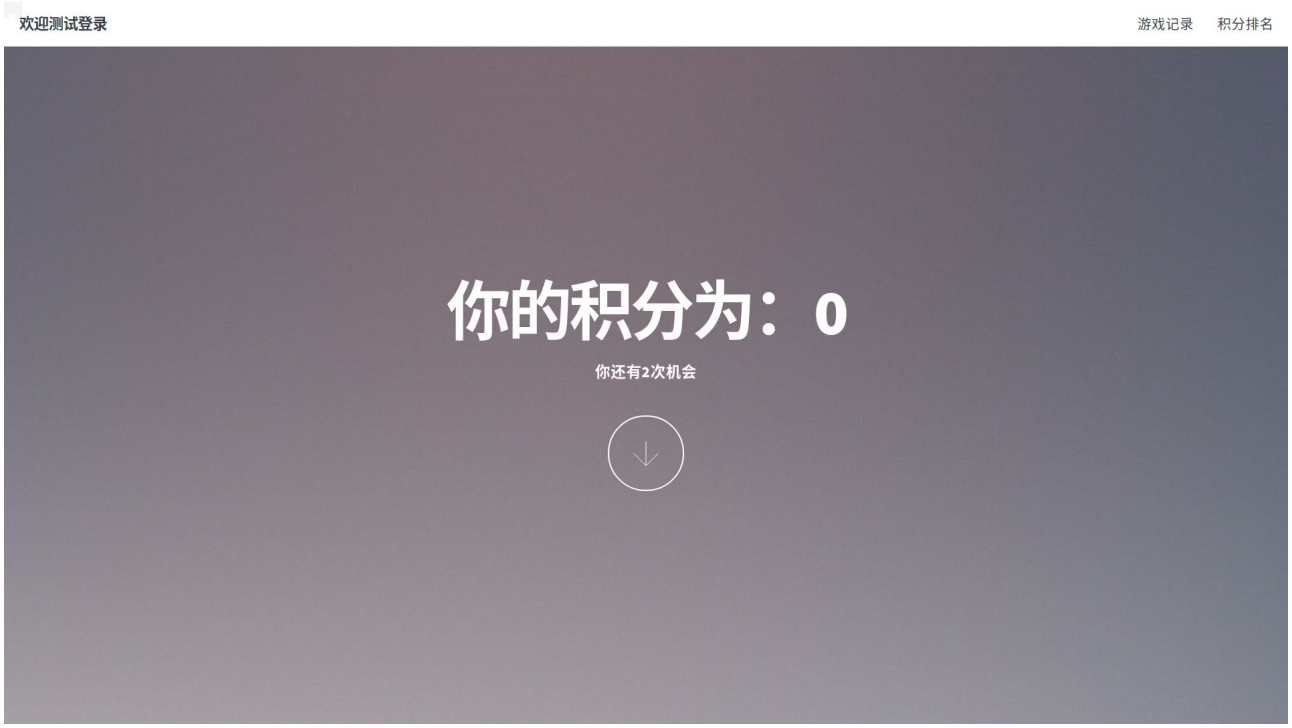
修改密码界面。直接输入新的密码即可。提交后会跳转到登录页面，需要重新登录。

选择难度后即可开始游戏，点击箭头进入答题。右上角可以查看本局游戏的排名与历史游戏记录。



随机生成一个单词与三个不重复的错误选项。根据用户选择由后台判断是否正确，如果超时或选错，判断为错误。

选择错误或超时，则允许错误的剩余次数减一。



选择正确积分加一。

允许错误的剩余次数为零时，游戏结束，跳转到本局的积分排名界面。

在游戏积分排名界面可以查看个人的历史游戏记录，这里记录了本用户曾经所有选择错误或超时的单词。

以管理员账号登录后，跳转到单词库的页面。这里可以模糊查询、增加、修改、删除单词。

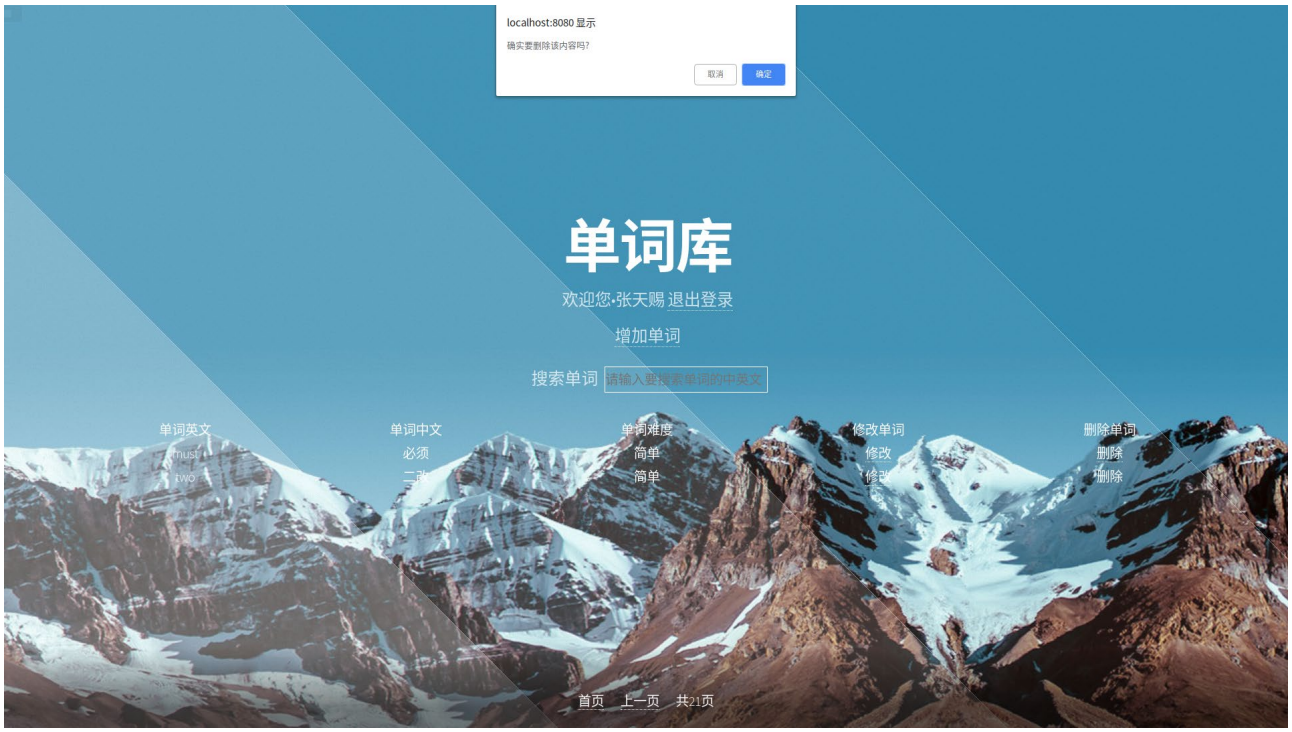
可以模糊查询单词的中、英文。我们认为 15 条以后的结果没有意义或意义很小，所以只显示模糊查询的前 15 条结果。

增加单词。填写单词的中文、英文和难度即可。

增加单词后，末页会显示出新增的单词。

修改单词。可以修改单词的英文、中文和难度。

删除单词。会弹出确认框，点击确认后删除。



5 课程设计出现的问题及解决的方法

前期在设计数据库表中，由于分析不充分，导致在后期的编码设计过程中遇到了问题，比如数据表冗余和缺失问题，后来在编码过程中重新设计了数据库表，解决了此类问题。

在任务完成后小组成员之间做一个总结，要熟悉整个工程的流程与步骤，对于不熟悉项目的组员要进行一个简单的将讲解。

6 课程设计的体会及自我评价与总结

经过为期两周的开发过程，小组成员通力合作，终于实现了老师要求开发的工程，完成了整个项目。

这次课程设计我们做的是英文单词游戏设计与实现。系统中采用的技术和环境主要有：JDBC 数据库连接技术、MYSQL8.0、Servlet、Ajax、Cookie、Filter、JavaScript、HTML5、CSS3、IntelliJ IDEA 集成开发环境。在项目的开发初期由于对项目开发的流程不熟悉，导致进程过于缓慢，遇到了很多问题，有的是知识存储不足，有的是考虑不够周全。其次，由于对需求分析不够充分仔细，导致了数据库冗余，在编码过程中增加了不少难度。开发过程中每位小组成员都遇到了不同的问题，但小组成员的开发激情很高，在所掌握的技术知识不

够的情况下，各成员相互支持，努力学习相关的知识，最终在团队所有成员的共同努力下，这些问题全部都迎刃而解。小组的每位成员都认真负责的对待自己开发的模块，终于在规定的时间内完成了老师安排的任务。

通过此次课程设计，使我们更加扎实的掌握了有关 JavaWeb 方面的知识，在设计过程中虽然遇到了一些问题，但经过一次又一次的思考，一遍又一遍的检查终于找出了原因所在，也暴露出了前期我们在这方面的知识欠缺和经验不足。在这次项目的开发中，小组中的每个人都了解到了团队合作的重要性，以及一个清晰的开发思路对项目的重要性，在开发过程中由于个人的不谨慎导致的 BUG 也有很多，所以在今后的学习与开发中要谨慎的对待每一次工作。我们不可能做到面面俱到，但一定要做到步步扎实，作为一个程序编程人员，要保持清醒的头脑，以现实为依据，让自己的每一行代码都能实现自己的意义。通过这次课程设计，我收获的不仅仅是课程上的知识得到实际应用，还有编程的基本习惯和开发系统时应注意的流程。并且也明白了项目开发经验对项目开发的重要性，在今后的应当多多进行项目开发，丰富经验。

实践出真知，通过亲自动手制作，使我们掌握的知识不再是纸上谈兵。过而能改，善莫大焉。在课程设计过程中，我们不断发现错误，不断改正，不断领悟，不断获取。在今后社会的发展和学习实践过程中，一定要不懈努力，不能遇到问题就想到要退缩，一定要不厌其烦的发现问题所在，然后一一进行解决，只有这样，才能成功的做成想做的事，才能在今后的道路上劈荆斩棘，而不是知难而退，那样永远不可能收获成功，收获喜悦，也永远不可能得到社会及他人对你的认可！课程设计诚然是一门专业课，给我很多专业知识以及专业技能上的提升，同时又是一门讲道课，一门辩思课，给了我许多道，给了我很多思，给了我莫大的空间。同时，设计让我感触很深。使我对抽象的理论有了具体的认识。我认为，在这学期的实验中，不仅培养了独立思考、动手操作的能力，在各种其它能力上也都有了提高。更重要的是，在实验课上，我们学会了很多学习的方法。而这是日后最实用的，真的是受益匪浅。要面对社会的挑战，只有不断的学习、实践，再学习、再实践。这对于我们的将来也有很大的帮助。以后，不管有多苦，我想我们都能变苦为乐，找寻有趣的事情，发现其中珍贵的事情。就像中国提倡的艰苦奋斗一样，我们都可以在实验结束之后变的更加成熟，会面对需要面对的事情。

7 参考文献

[1] 田保军，刘利民. 软件工程实用教程[M] . 北京：清华大学出版社，2015

[2] 马志强，张然，李雷孝. Java 核心技术[M] . 北京：清华大学出版社，2014

[3] 高凯，张雪梅，白云飞. 数据库原理与应用[M]. 北京：电子工业出版社，2011

8 附录

8.1 PlayServlet(实现游戏功能)

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
            //处理编码
            request.setCharacterEncoding("utf-8");

            //获取 type
            String type = request.getParameter("type");

            //获取难度
            String Difficulty = request.getParameter("Difficulty");

            WordDao wd = new WordDao();
            UserDao ud = new UserDao();

            //获取 id
            String uid = request.getParameter("id");

            //根据 id 查询用户
            User u = ud.selectUserById(uid);

            //生成题目
            List<Word> listtrue = wd.selectOne(Difficulty);
            List<Word> listflase = wd.selectThree();
            List<Word> listall = new ArrayList<Word>();
            listall.addAll(listtrue);
            listall.addAll(listflase);

            //打乱 list
            Collections.shuffle(listall);
```

```
//初始化生成四个单词的数据
if ("play".equals(type)) {
    //初始化允许的错误次数
    int sum = 3;

    NotesDao nd = new NotesDao();
    Notes n = new Notes();

    /*//生成一条初始记录
    n.setUid(u.getId());
    n.setWid(listtrue.get(0).getId());
    n.setFrequency(0);

    //添加记录
    nd.insertOne(n);*/

    //初始化积分
    ud.updateDifficulty(u, 0, Difficulty);

    //查看单词数据
    /*for (int i = 0; i < listall.size(); i++) {
        System.out.println(listall.get(i));
        System.out.println(i);
    }*/

    //题目生成成功
    if (listtrue != null && listflase != null) {
        //传值
        PlayPassingValues P = new PlayPassingValues();
        P.playPassingValues(request,                    response,
ud.selectDifficulty(uid, Difficulty), listtrue, listall, u, sum, Difficulty);
```

```
            }
            //判断对错
        } else if ("judge".equals(type)) {
            //获取用户选择的汉语 id，正确答案的 id，剩余次数，单词难度（随机难度需要用）
            int fid = Integer.parseInt(request.getParameter("fid"));
            int tid = Integer.parseInt(request.getParameter("tid"));
            int sum = Integer.parseInt(request.getParameter("sum"));
            String dif = request.getParameter("dif");

            //选对
            if (fid == tid) {
                if (Difficulty.equals("随机")) {
                    if (dif.equals("简单")) {
                        //积分加一，更新数据库
                        ud.updateDifficulty(u,       ud.selectDifficulty(uid,
Difficulty) + 1, Difficulty);
                    } else if (dif.equals("中等")) {
                        //积分加二，更新数据库
                        ud.updateDifficulty(u,       ud.selectDifficulty(uid,
Difficulty) + 2, Difficulty);
                    } else if (dif.equals("困难")) {
                        //积分加三，更新数据库
                        ud.updateDifficulty(u,       ud.selectDifficulty(uid,
Difficulty) + 3, Difficulty);
                    }
                } else {
                    //积分加一，更新数据库
                    ud.updateDifficulty(u,           ud.selectDifficulty(uid,
Difficulty) + 1, Difficulty);
                }
                NotesDao nd = new NotesDao();
```

```
                //查询该用户是否有这个单词的游戏记录
                Notes n = nd.selectNotes(uid, request.getParameter("tid"));

                //如果有 正确次数加 1
                if (n != null) {
                    nd.updateOnee(n, (n.getTrequency() + 1));
                    //如果没有 生成一条记录
                } else {
                    //添加记录
                    Notes nn = new Notes();
                    nn.setUid(u.getId());
                    nn.setWid(tid);
                    nn.setFrequency(0);
                    nn.setTrequency(1);
                    nd.insertOne(nn);
                }

                //题目生成成功
                if (listtrue != null && listflase != null) {
                    //传值
                    PlayPassingValues P = new PlayPassingValues();
                    P.playPassingValues(request,                    response,
ud.selectDifficulty(uid, Difficulty), listtrue, listall, u, sum, Difficulty);
                }
                //选错
            } else {
                //允许的错误次数减 1
                sum--;
                NotesDao nd = new NotesDao();

                //查询该用户是否有这个单词的游戏记录
```

```
Notes n = nd.selectNotes(uid, request.getParameter("tid"));

//如果有 错误次数加 1
if (n != null) {
    nd.updateOne(n, (n.getFrequency() + 1));
    //如果没有 生成一条记录
} else {
    //添加记录
    Notes nn = new Notes();
    nn.setUid(u.getId());
    nn.setWid(tid);
    nn.setFrequency(1);
    nn.setTrequency(0);
    nd.insertOne(nn);
}

//如果允许的错误次数为 0 游戏结束
if (sum <= 0) {
    //生成用户积分排名
    List<User> list = ud.rankAll(Difficulty);

    //积分排名生成成功
    if (list != null) {
        //分页
        Pagination p = new Pagination();
        p.pagination(request);

        //获取总页数
        int countPage = ud.selectNum() / 15 + 1;

        //传值
        ScorePassingValues S = new ScorePassingValues();
```

```
                        S.scorePassingValues(request, response, countPage,
list, u, Difficulty);
                    }
                    //如果允许的错误次数大于 0 游戏继续
                } else {
                    //题目生成成功
                    if (listtrue != null && listflase != null) {
                        //传值
                        PlayPassingValues P = new PlayPassingValues();
                        P.playPassingValues(request, response,
ud.selectDifficulty(uid, Difficulty), listtrue, listall, u, sum, Difficulty);
                    }
                }
            }
            //再来一局
        } else if ("oneMoreGame".equals(type)) {
            request.setAttribute("user", u);
            request.getRequestDispatcher("play-welcome.jsp").forward(request,
response);
        }
    }
```

8.2 rankAll(生成积分排名)

```
    ublic List<User> rankAll(String difficulty) {
        try {
            //1.得到一个连接
            Connection conn = DBUtil.getConn();
            //2.得到操作数据库对象
            String sql = "SELECT *,(SELECT count(DISTINCT "+difficulty+") FROM
用户 AS b WHERE a."+difficulty+"<b."+difficulty+" AND 用户序号>1)+1 AS 累计积分排
名 FROM 用户 AS a WHERE 用户序号>1 ORDER BY 累计积分排名";
            //String sql = "SELECT CASE WHEN @prevRank = " + difficulty + " THEN
@rank WHEN @prevRank := " + difficulty + " THEN @rank := @rank+1 END AS 累计积分
```

```
排名,a.* FROM(SELECT * FROM 用户)a,(SELECT @rank :=0,@prevRank :=NULL)b ORDER BY a." + difficulty + " DESC";
                PreparedStatement stmt = conn.prepareStatement(sql);
                ResultSet rs = stmt.executeQuery();//执行
                List<User> list = new ArrayList<User>();
                //3.bean 对象封装
                while (rs.next()) {
                    User user = new User();
                    user.setId(rs.getInt("用户序号"));
                    user.setName(rs.getString("账号"));
                    user.setPwd(rs.getString("密码"));
                    user.setSimpleScore(rs.getInt("简单"));
                    user.setMediumScore(rs.getInt("中等"));
                    user.setHardScore(rs.getInt("困难"));
                    user.setRandom(rs.getInt("随机"));
                    user.setRanking(rs.getInt("累计积分排名"));
                    list.add(user);
                }
                rs.close();
                stmt.close();
                conn.close();
                return list;
            } catch (Exception e) {
                e.printStackTrace();
                return null;
            }
        }
```