1. **数据库设计**

1.数据库的必要性和可行性分析

我们考虑到，在本实验中，可能用得到数据库的部分有以下几个方面，如用户的注册和登录功能，机器人的管理功能，交互式聊天记录需要缓存。遵循奥卡姆剃刀法则，我们应对可能出现的问题进行一些简化，并为不同的情况选择最优的解决方案。经过与其他后端同学的讨论，我们认为聊天记录可以使用HTML5的特性进行离线缓存，而且用数据库设计聊天记录的保存十分复杂且浪费系统资源。所以，我们为用户的注册和登录和机器人管理设计数据库。在多种开源数据库中，SQLITE3对JavaScript没有提供原生支持，考虑可行性，所以我们最终选择了MySQL的方案来进行数据库的设计。

2.注册的功能流程

A.用户首先输入账户名和密码，输入应为合法的字符串，两次密码应一致

B.在填写完成后，点击注册

C.如果用户的账户名没有被注册过，且两次输入的密码一致，那么为其分配一个唯一的ID。否则，提示用户重新注册。

注册的数据流程

A.首先，查询用户提交的用户名，

B.根据用户名是否存在和两次密码验证相同，判断用户是否注册成功

C.如果注册成功，则对数据库进行更新，并为其分配一个唯一的id，储存于数据库中

3.登录的功能流程

A.用户首先输入账户名和密码

B.在填写完成后，点击登录

C.如果用户的账户名没有被注册过，则登录失败。如果注册过，且验证成功，则登录成功。

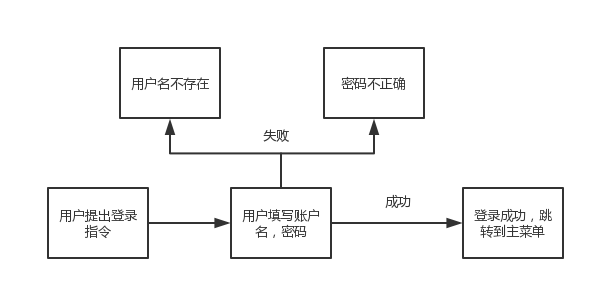
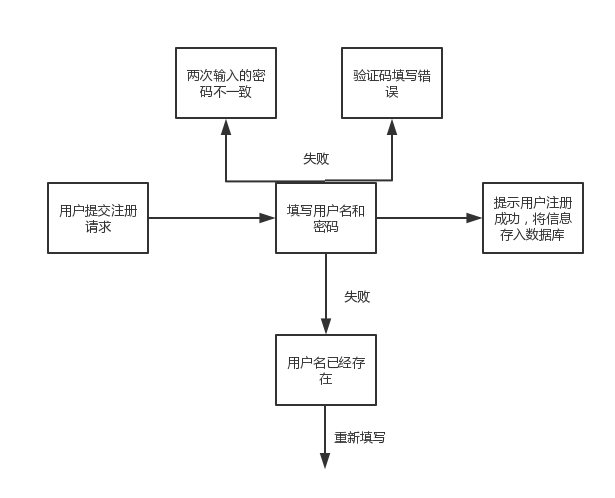
登录的数据流程

A.首先，查询用户提交的用户名，

B.如果用户名存在，则在数据库中验证密码，验证成功则登录成功。

C.如果用户名不存在或密码不一致，则登录失败

4.逻辑流程图



4.设计阶段

在用户登录阶段，总共设计一个栏位，储存用户的用户名，密码，和邮箱。理所应当的，设置这几个项都为NOT FULL。并参考以下设计原则：

（一）降低对数据库功能的依赖

　　功能应该由程序实现，而非DB实现。原因在于，如果功能由DB实现时，一旦更换的DBMS不如之前的系统强大，不能实现某些功能，这时我们将不得不去修改代码。所以，为了杜绝此类情况的发生，功能应该有程序实现，数据库仅仅负责数据的存储，以达到最低的耦合。

（二）定义实体关系的原则

　　当定义一个实体与其他实体之间的关系时，需要考量如下：

牵涉到的实体 识别出关系所涉及的所有实体。

所有权 考虑一个实体“拥有”另一个实体的情况。

基数 考量一个实体的实例和另一个实体实例关联的数量。

　　关系与表数量

描述1:1关系最少需要1张表。

描述1:n关系最少需要2张表。

描述n:n关系最少需要3张表。

（三）列意味着唯一的值

　　如果表示坐标（0,0），应该使用两列表示，而不是将“0,0”放在1个列中。

（四）列的顺序

　　列的顺序对于表来说无关紧要，但是从习惯上来说，采用“主键+外键+实体数据+非实体数据”这样的顺序对列进行排序显然能得到比较好的可读性。

（五）定义主键和外键

　　数据表必须定义主键和外键（如果有外键）。定义主键和外键不仅是RDBMS的要求，同时也是开发的要求。几乎所有的代码生成器都需要这些信息来生成常用方法的代码（包括SQL文和引用），所以，定义主键和外键在开发阶段是必须的。之所以说在开发阶段是必须的是因为，有不少团队出于性能考虑会在进行大量测试后，在保证参照完整性不会出现大的缺陷后，会删除掉DB的所有外键，以达到最优性能。笔者认为，在性能没有出现问题时应该保留外键，而即便性能真的出现问题，也应该对SQL文进行优化，而非放弃外键约束。

5.实现阶段

Node.js 配置路由与数据库建立连接：

文件位置：\bots\node\_modules\dao\dbConnect.js

var mysql=require('mysql');

function connectServer(){//设置服务器连接

var client=mysql.createConnection({

host:'localhost',//本地服务器

user:'root',//根权限

password:'',//为了测试的稳定性密码为空

database:'botsusers'//数据库端为工程建立“botsusers“数据库，用于存放//用户注册信息

})

return client;//建立连接成功则正常返回

}

function selectFun(client,username,callback){

//client为一个mysql连接对象，查找用户名是否已经存在

client.query('select password from userinfo where username=//数据库查找语句

"'+username+'"',function(err,results,fields){

if(err) throw err;

callback(results);

});

}

function insertFun(client , username , password,mail,callback){

client.query('insert into userinfo value(?,?,?)', [username, password,mail], function(err,result){//若用户名不存在则可以注册，插入表项，name password email

if( err ){

console.log( "error:" + err.message);

return err;//异常检测

}

callback(err);

});

}

//指定当前对象的引用

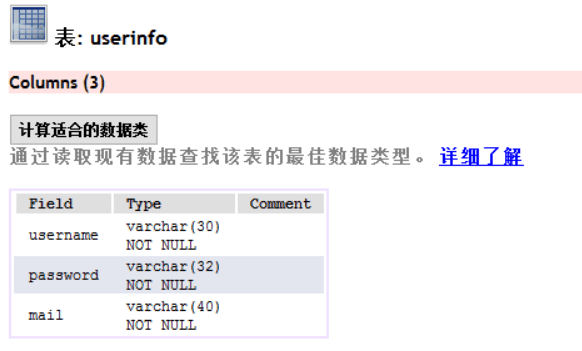
exports.connect = connectServer;

exports.selectFun = selectFun;

exports.insertFun = insertFun;

数据库列表项截图：





1. **基于深度学习的自然语言处理系统**
2. 中文语料收集
3. **维基百科**

优点：

* 维基百科资源获取非常方便，有[Wiki Dump](https://dumps.wikimedia.org/zhwiki/latest/)可以直接下载。其他可能需要用爬虫抓取或者付费。
* 维基百科的文档解析有非常多的成熟工具，直接使用开源工具即可完成正文的提取。
* 维基百科的质量较高，而且领域广泛。

缺点：

* 最主要的就是数量较少，相比国内的百度百科、互动百科等，数据量要少一个数量级。

1. 新闻稿件

优点：

* 资料获取容易
* 具有权威性

缺点：

* 根据实际训练结果，基于新闻的处理结果对自然语言的理解并不理想
* 无法清除识别口语化的表达

1. 语料库（corpus）

语料库（corpus）一词在语言学上意指大量的文本，是在语言的实际使用中真实出现过的语言材，通常经过整理，具有既定格式与标记。国内较权威的语料库有以下几个：

1国家语委现代汉语语料库<http://www.cncorpus.org/>

2《人民日报》标注语料库<http://www.icl.pku.edu.cn/icl_res/>

3语言研究所：北京口语语料查询系统（B J K Y）<http://www.blcu.edu.cn/yys/6_beijing/6_beijing_chaxun.asp>

4现代汉语平衡语料库<http://www.sinica.edu.tw/SinicaCorpus/>

5在线分词标注系统<http://ling.cuc.edu.cn/cucseg/>

6哈工大信息检索研究室对外共享语料库资源<http://ir.hit.edu.cn/demo/ltp/Sharing_Plan.htm>

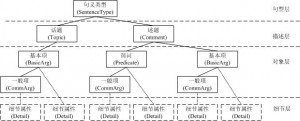
优点：

* 具有良好的自然语言理解能力。
* 不用去筛选数据，可以直接使用数据集进行训练。

缺点：

* 无法自定义数据集，特殊预警的泛化能力较差

4）句义结构的基本形式



可见，我们为了较好的效果，最终的语料集应尽量满足这个结构。

5）最终语料的方案

出于多方面的考虑，我们决定选择国内一知名语料库中的中文对白语料，作为我们的训练和测试数据集。对其进行观察，我发现对白分别取自于不同的资料；根据我们的观察，原始数据全部来自电视剧的中文语料，经过人工收集、整理，合理覆盖了主谓句、非主谓句、把字句等六种主要句式类型，规模已达万句。

1. 分词系统
2. 分词方案

中文分词算法大概分为两大类

a.第一类是基于字符串匹配，即扫描字符串，如果发现字符串的子串和词相同，就算匹配。

这类分词通常会加入一些启发式规则，比如“正向/反向最大匹配”, “长词优先” 等策略

这类算法优点是速度块，都是O(n)时间复杂度，实现简单，效果尚可。  
也有缺点，就是对歧义和未登录词处理不好。

歧义的例子很简单"长春市/长春/药店" "长春/市长/春药/店".  
未登录词即词典中没有出现的词，当然也就处理不好。

ikanalyzer,paoding 等就是基于字符串匹配的分词。

b.第二类是基于统计以及机器学习的分词方式

这类分词基于人工标注的词性和统计特征，对中文进行建模，即根据观测到的数据（标注好的语料）对模型参数进行估计，即训练。 在分词阶段再通过模型计算各种分词出现的概率，将概率最大的分词结果作为最终结果。常见的序列标注模型有HMM和CRF。

这类分词算法能很好处理歧义和未登录词问题，效果比前一类效果好，但是需要大量的人工标注数据，以及较慢的分词速度.

1. 分词粒度

这里有一个真实的例子，即  
"团购网站的本质是什么？"  
这是一个知乎的问题，如果使用单一粒度的分词，如果我们有  
  
  
团购  
团购网   
网站   
团购网站   
本质   
是   
什么

这些词在词典中，按最大匹配分词结果是  
“团购网站/的/本质/是/什么”   
  
  
当用户输入  
“团购网的本质”   
分词结果是  
“团购网/的/本质”   
“团购网”这个词显然是没有匹配的。

同样，如果按最小匹配分词，会有类似的问题。

因此，我们考虑基于字符串匹配的分词方法最好能够匹配出多粒度的结果，即  
能分出  
“团购网站/团购/团购网/网站/的/本质/是/什么”   
这样多粒度的结果。

所以为了达到好的效果，我们最终选了ik-analyzer方案，IK Analyzer是一个开源的，基于java语言开发的轻量级的中文分词工具包。IK-analyzer采用了特有的“正向迭代最细粒度切分算法“，支持细粒度和智能分词两种切分模式

1. 准确率和效果

**测试文本1**

SCANV网址安全中心([http://scanv.com](https://link.zhihu.com/?target=http%3A//scanv.com))是一个综合性的网址安全服务平台。通过网址安全中心，用户可以方便的查询到要访问的网址是否存在恶意行为，同时可以在SCANV中在线举报曝光违法恶意网站。

**测试结果：**

SCANV/x 网址/n 安全/an 中心/n (/wkz [http://scanv.com/x](https://link.zhihu.com/?target=http%3A//scanv.com/x) )/wky 是/vshi 一个/mq 综合性/n 的/ude1 网址/n 安全/an 服务平台/n 。/wj 通过/p 网址/n 安全/an 中心/n ，/wd 用户/n 可以/v 方便/a 的/ude1 查询/vn 到/v 要/v 访问/v 的/ude1 网址/n 是否/v 存在/v 恶意/n 行为/n ，/wd 同时/c 可以/v 在/p SCANV/x 中/f 在/p 线/n 举报/vn 曝光/vn 违法/vn 恶意/n 网站/n 。/wj

**测试文本2：**

随着页游兴起到现在的页游繁盛，依赖于存档进行逻辑判断的设计减少了，但这块也不能完全忽略掉。总会有一些功能是需要调用本地存档的。例如登录模块中，记住密码功能，会将密码信息存储在本地，以IE浏览器为例，在C:\Documents and Settings\（你的Windows用户名）\Application Data\Macromedia&nbsp;\Flash Player\#SharedObjects\（一些随机数字和字母）\ 文件夹下就可以看到存储密码的SOL文件，可以使用minerva工具查看，如下图所示，密码明文明文存储的，SOL文件是永久性保存的，除非手动清除，如果玩家在公共环境下登录，就会有盗号威胁。来自[http://www.baidu.com/](https://link.zhihu.com/?target=http%3A//www.baidu.com/)及http：、[www.baidu...com](http://www.baidu...com)

**测试结果：**

随着/p 页/q 游兴/n 起/vf 到/v 现在/t 的/ude1 页/q 游/v 繁盛/an ，/wd 依赖/v 于/p 存档/vi 进行/vx 逻辑/n 判断/v 的/ude1 设计/vn 减少/v 了/y ，/wd 但/c 这/rzv 块/q 也/d 不能/v 完全/ad 忽略/v 掉/v 。/wj 总/d 会/v 有/vyou 一些/mq 功能/n 是/vshi 需要/v 调用/v 本地/rzs 存档/vi 的/ude1 。/wj 例如/v 登录/v 模块/n 中/f ，/wd 记住/v 密码/n 功能/n ，/wd 会/v 将/p 密码信息存储/n 在/p 本地/rzs ，/wd 以/p IE/x 浏览器/n 为/p 例/n ，/wd 在/p C:/x \/x Documents/x /w and/x /w Settings/x \/x （/wkz 你/rr 的/ude1 Windows/x 用户/n 名/q ）/wky \/x Application/x /w Data/x \/x Macromedia/x &/x nbsp/x ;/wf \/x Flash/x /w Player/x \/x #/x SharedObjects/x \/x （/wkz 一些/mq 随机/b 数字/n 和/cc 字母/n ）/wky \/x /w 文件夹/n 下/f 就/d 可以/v 看到/v 存储/vn 密码/n 的/ude1 SOL/x 文件/n ，/wd 可以/v 使用/v minerva/x 工具/n 查看/v ，/wd 如/v 下/vf 图/n 所/usuo 示/vg ，/wd 密码/n 明/ag 文明/n 文/ng 存储/v 的/ude1 ，/wd SOL/x 文件/n 是/vshi 永久性/n 保存/v 的/ude1 ，/wd 除非/c 手动/b 清除/vn ，/wd 如果/c 玩/v 家/n 在/p 公共/b 环境/n 下/f 登录/v ，/wd 就/d 会/v 有/vyou 盗/vg 号/n 威胁/vn 。/wj 来自/v http:/x //w //w [http://www.baidu.com/x](https://link.zhihu.com/?target=http%3A//www.baidu.com/x) //w 及/v http：/x 、/wn www.baidu...com/x

**分析：**

经过样例测试，该分词系统可以基本满足我们的需要。性能方面，可以达到30左右的交叉歧义错误率，并且，每秒可处理约15万字。

1. 语义分析
2. 使用的技术（包括框架）
3. 各个module的设计

参考资料：

Ik-analyzer: https://code.google.com/archive/p/ik-analyzer/