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

D.如果用户名重复，则跳转到“用户名已存在”界面

**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。

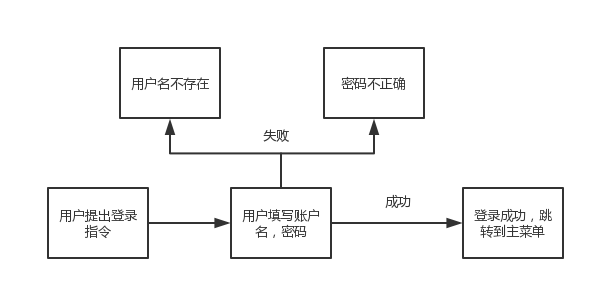
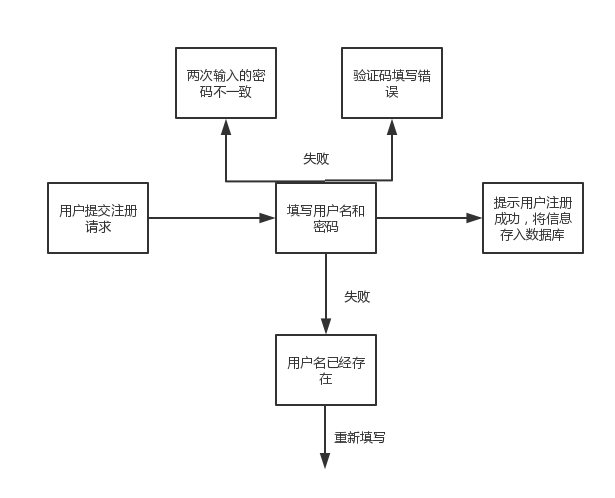
登录的数据流程

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**



**5 .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(username)，密码(password)，和邮箱(email)。理所应当的，设置这几个项都为NOT FULL。并参考以下设计原则进行设计：

（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。在本数据库设计中，我们将用户名(username)，密码(password)，和邮箱(email)分别存在三个列中。

（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。在本数据库设计中，我们将用户名（username），密码(password)，和邮箱（email）中的与户名（username）定义为主键，并

（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。在本数据库设计中，我们将用户名（username），密码(password)，和邮箱（email）中的用户名（username）定义为主键，密码（password）和邮箱（email）设置为外键。并且另用户名(username)不可重复（unique），并且在数据库中进行设置。

**6. 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*

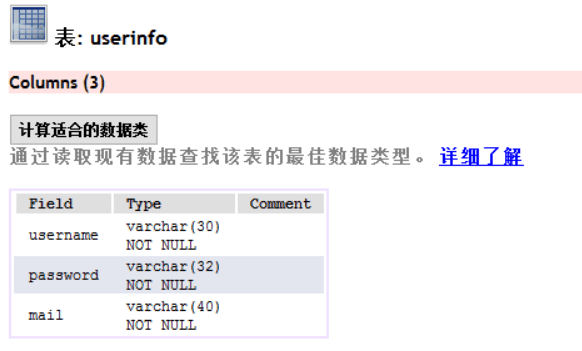
exports.connect = connectServer;

exports.selectFun = selectFun;

exports.insertFun = insertFun;

数据库列表项截图：





//数据库跳转

//设计更改