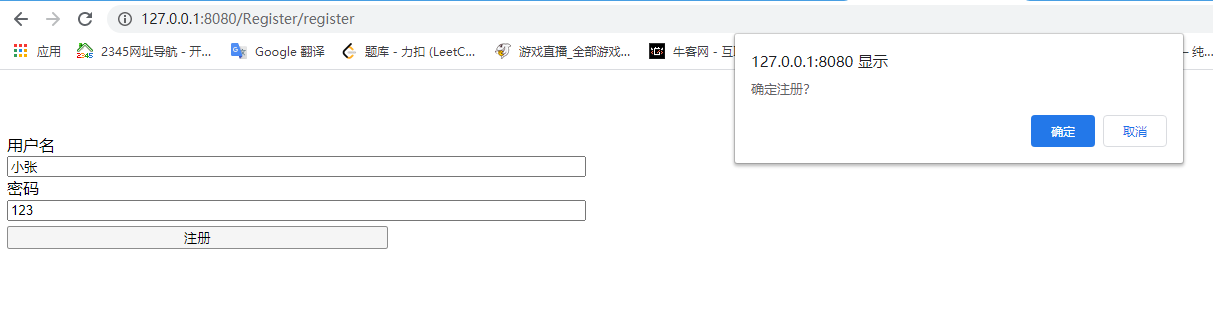
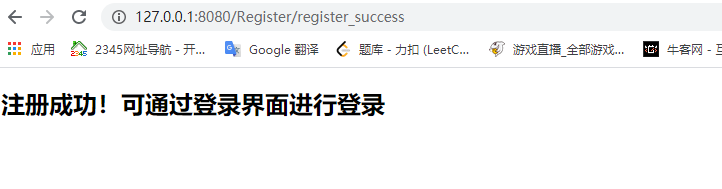
# 1 注册

## 1.1 注册界面与注册成功界面





## 1.2 前端ajax请求代码

|  |
| --- |
| function fuc() {  var r = confirm("确定注册？");  if (r == true) {  var username = $("#username").val();  var password = $("#password").val();  var ajaxParams = {};  ajaxParams["username"] = username;  ajaxParams["pwd"] = password;  $.post("../Register/user\_inserttodb?random=" + Math.random(), ajaxParams, function (response) {  //alert(response);  var testJson = eval(response);  if (testJson[0]["msg"] == 'success') {  alert("注册成功！");  window.location.href = "../Register/register\_success";  }  else {  alert('注册失败！');  }  });  } } |

## 1.3 数据实体

|  |
| --- |
| @Data public class User {  private String id;  private String userName;  private String pwd; } |

## 1.4 控制器

|  |
| --- |
| @RequestMapping("/user\_inserttodb") @ResponseBody public String user\_inserttodb(HttpServletRequest request)throws ServletException,IOException {  String username = request.getParameter("username");  String pwd = request.getParameter("pwd");  SimpleDateFormat df = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");//设置日期格式;  String id=df.format(new Date()).toString();  String uu="";  try  {  registerMapper.insertDb(id, username, pwd);  uu = "[{msg:'success'}]";  }  catch (Exception e)  {  uu = "[{msg:'fail',failInfo:'" + e.toString() + "'}]";  }  return uu; } |

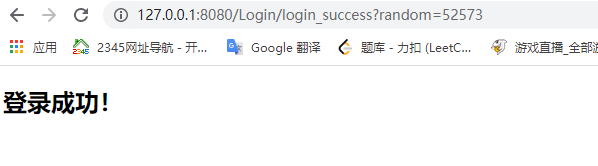
## 1.5数据持久化层

|  |
| --- |
| @Repository @Mapper public interface RegisterMapper {  @Insert("INSERT INTO USER(id, username, pwd) VALUES( #{id}, #{userName}, #{pwd})")  void insertDb(@Param("id") String id,  @Param("userName") String userName,  @Param("pwd") String pwd); } |

# 2 登录

## 2.1 登录界面与登录成功界面





## 2.2 前端ajax请求

|  |
| --- |
| function login() {  var password= document.getElementById("password").value;  var username = document.getElementById("username").value;  $.ajax({  type: "post",  url: "../Login/login\_check",  dataType: "text",  async: false,  data: {  userName: username,  password: password  },  success: function (response) {  var testJson = eval(response);  if (testJson[0]["msg"] == 'success') {  window.location.href = "../Login/login\_success?random=" + Math.floor(Math.random() \* (100000 + 1));  }  else {  alert(testJson[0]["failInfo"]);  document.getElementById("username").value = "";  document.getElementById("password").value = "";  }  }  }); } |

## 2.3 数据实体

|  |
| --- |
| @Data public class User {  private String id;  private String userName;  private String pwd; } |

## 2.4 控制器

|  |
| --- |
| @ResponseBody @RequestMapping(value="/login\_check",method= RequestMethod.POST) public String login\_check(HttpServletRequest request)throws ServletException,IOException {  String res = "";  String userName = request.getParameter("userName");  String password = request.getParameter("password");  List<User> list=new ArrayList<User>();  list=loginMapper.LoginDao(userName,password);  if(list.size()==0)  {  res = "[{msg:'fail',failInfo:'账号密码认证失败，请重试 '}]";  }else{  String username=list.get(0).getUserName();  res = "[{msg:'success',username:'" + username + "'}]";  }  return res; } |

## 2.5 数据持久化层

|  |
| --- |
| @Repository @Mapper public interface LoginMapper {  @Select("SELECT \* FROM USER WHERE username = #{userName} AND pwd=#{pwd}")  List<User> LoginDao(@Param("userName")String userName, @Param("pwd")String pwd); } |