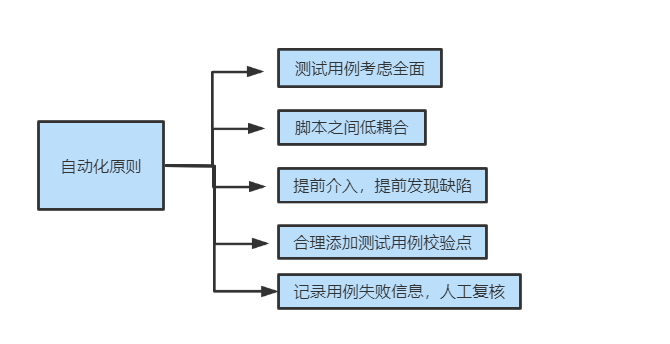


招聘岗位手册

# 招聘原则



# 招聘人员胜任素质模型

**知识**

数据库

Linux 操作系统

Python语言

Jenkins 持续集成

**技能**

数据库，linux基本操作

Python 自动化脚本的编写

设计测试用例

Jenkins持续集成

沟通表达能力

团队合作能力

**素养**

原则性

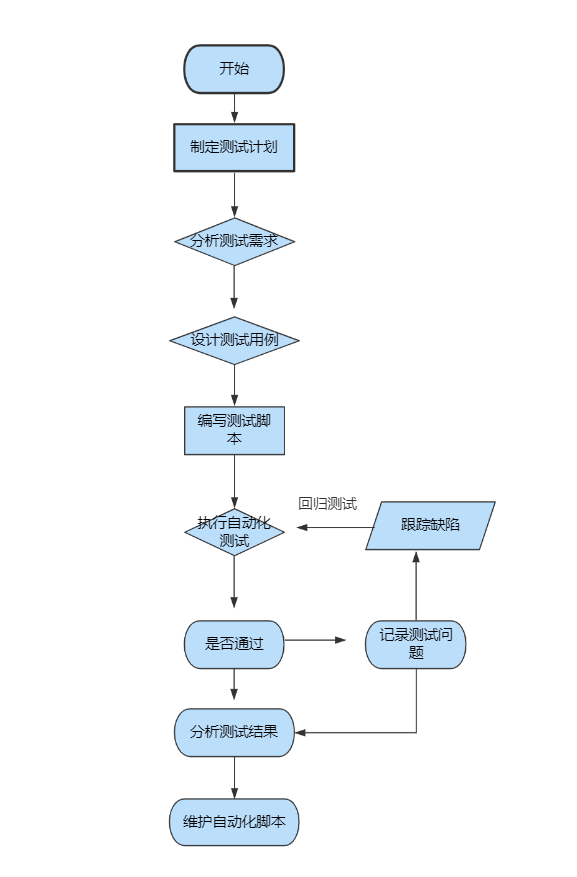
自信心

保密意识

积极主动

抗压能力强

# 工作流程图



# 自动化实施

## 制定测试计划

### **目的**

让整个项目组成员，包括测试人员，开发人员，技术文档作者，管理部门等，了解、同意以及支持测试计划的执行;

测试的目的是项目按照测试计划如期执行，项目按时上线。

### **定义**

对于文件中的专业术语，需要项目全部成员达成一致。（软件未实现产品说明书要求的功能;软件中出现了产品说明书指明不应该出现的错误；软件实现了产品说明书中未提到的功能；软件未实现产品说明书虽未明确提及但应该实现的目标）

### **明确测试内容**

根据项目的需求文档，明确需要自动化测试的模块；不能使用自动化测试的模块，需给出具体的原因。

### **测试的阶段**

根据开发情况制定每一个阶段的测试任务，并将测试任务执行情况告知项目组，以便于项目的整体进度的推进。

### **测试策略**

主要描述测试小组用于测试过程中采用的测试策略的方法，如：自动化测试是接口自动化还是UI自动化？自动化测试怎么与功能测试相配合，以便减少功能测试中的重复性操作工作。

### **资源需求**

人员，设备，办公室和实验室空间，软件，其它设备等

### **任务分配**

明确测试人员负责软件的哪些模块，确保软件的每一部分都分配有人测试；每一个测试员清楚地知道自己负责的模块；在每一个测试阶段，自己的具体任务是什么。

### **进度安排**

进行测试进度安排，可以为项目组人员，项目经理提供信息，以便更好的安排整个项目的进度;其中会出现“进度破坏”的情况，此时测试进度可采用相对日期的方法，依赖于其它先完成的可交付内容。

### **软件缺陷报告**

使用什么方式记录缺陷，从发现到修复的过程中如何去跟踪？

### **度量和统计**

需明确收集哪些信息，要做什么决定，谁来负责收集,如:

1. 待修复的缺陷清单
2. 当前软件的缺陷严重程度分布
3. 自动化脚本执行过程中发现的缺陷的数量
4. 缺陷发布的趋势和区域

### **风险和问题**

明确指出项目潜在问题或者风险区域，并在计划中进行列出，在进度中给予说明。

最后注意的是，测试计划不是一成不变的，随着项目的进行，会由于各方面的因素（如：提交测试的程序版本质量低、bug量大修改慢、需求变更等等）导致测试计划无法按原计划执行，这时要适当的调整测试计划。

## 分析测试需求

用例设计者根据测试计划和需求说明书，分析测试需求，设计测试需求树，以便用例设计时能够覆盖所有的需求点。一般来讲，基于Web功能测试需要覆盖一下几个方面：

1. 页面链接测试，确保各个链接正常；
2. 页面控件测试，确保各个控件可靠；
3. 页面功能测试，确保各项操作正常；
4. 数据处理测试，确保数据显示准确、处理精确可靠；
5. 模块业务逻辑测试，确保各个业务流程畅通。

## 设计测试用例

通过分析测试需求，设计出能够覆盖所有需求点的测试用例，形成专门的测试用例文档。由于不是所有的测试用例都能用自动化来执行，所以需要将能够执行自动化测试的用例汇总成自动化测试用例。必要时，要将登陆系统的用户、密码、产品、客户等参数信息独立出来形成测试数据，便于脚本开发。

常用的测试方法有边界值，场景法，等价类划分法等

### 边界值

对输入的边界值或稍大（小）于边界值的值进行分析。比如对某一输入框的长度要求为10个字符，那么1、10的长度为真，0、11的长度为假

### 场景法

根据需求文档，梳理项目的整体流程。场景法一般分为基本流，备用流，异常流。

### 等价类划分法：

等价类是输入的集合，比如在注册时，密码规定为6-16位英文字母或数字及下划线，那么小于6位的一串字符就是一个等价类，大于16位的一串字符是另一个等价类，在6-16位之间且符合规范的一串字符也是一个等价类，在6-16位之间的但包含除英文字母和数字和下划线之外的字符是另外一个等价类。在每个等价类中选取一定数目的值作为代表。等价类分为有效等价类和无效等价类，输入符合条件的值对功能进行检验，输入无效等价类中的值可以找出程序错误的地方。

## 编写测试脚本

### 自动化的选择

根据项目的整体情况，各个模块之间的关联，决定是采用UI自动化还是接口自动化；对于项目的哪些模块用自动化来实现。

### 脚本编写

脚本的编写的原则有：

1. 测试代码与测试数据相分离；
2. 测试代码之间要减少相互之间的影响，要低耦合；
3. 测试代码之间的层级分明，代码格式规范，适当添加注释，便于他人理解；
4. 测试代码要便于维护。

## 分析测试结果、记录测试问题

应该及时分析自动化测试结果，建议测试人员每天抽出一定时间，对自动化测试结果进行分析，以便尽早地发现缺陷。

自动化测试案例运行结果后，发送对应的结果统计邮件到邮箱，测试人员只需根据统计的邮件，找到测试文件中运行失败的测试用例，复核测试结果是否有问题；确认是真实的系统缺，就提交开发人员修复，如果不是系统缺陷，就检查自动化测试脚本或者测试环境。

## 跟踪测试BUG

测试记录的BUG要记录到缺陷管理工具中去，以便定期跟踪处理。开发人员修复后，需要对此问题执行回归测试，就是重复执行一次该问题对应的用例，执行通过则关闭，否则继续修改。如果问题的修改方案与客户达成一致，但与原来的需求有所偏离，那么在回归测试前，还需要对脚本进行必要的修改和调试。

## 自动化脚本的维护

如果系统需求发生变更时，对自动化测试脚本和相关文档包括《自动化测试用例》、《自动化脚本设计说明书》进行维护，以适应变更后的系统。