功能自动化测试实施方案

**案例：中央财务理财系统自动化测试实施案例**

|  |  |
| --- | --- |
| 项目名称 | 中央财务理财系统 |
| 项目和功能描述 | 进行投资购买保险购买基金查看涨势和热门投资板块，系统优点：可以在线进行投资，也可以进行线上购买基金，可以查看到热点板块。保险的种类比线下的多，理财的种类也多样化，让客户能够更直观的了解到理财的优劣，比线下的理财讲解更好理解。平台为不同类型的用户提供投资理财产品，解决不同类型用户的理财需求。还与国内的多家基金机构有深入的合作，理财产品的价格亲民门槛低产品保真。  系统功能：主要功能：用户：个人理财管理：零钱理财，工资理财，期限理财，基金理财，银行推荐 金融工具：资金记录往来，网贷申请 个人中心：我的理财，我的借贷，银行卡管理，账户安全 管理员： 用户信息管理：用户信息，银行卡管理，个人征信 理财产品管理：零钱理财，工资理财，期限理财，基金理财，精选银行 权限管理：用户权限，管理员权限 网贷管理：网贷审核，网贷信息查看，到期提醒 |
| **前期：分析出哪些模块适合做自动化，哪些不适合？需要对模块进行划分。** | |
| 1. **脚本对用例的覆盖率：80~90%（指纹识别，验证码，人脸扫描，第三方硬件设备）**   **原因：当需求不完整时，测试很难实现自动化，无法提供准确的测试覆盖。**  **低优先级的需求可能不符合自动化测试的条件。这可以减少通过测试自动化的测试覆盖水平。**  **不同测试阶段的测试覆盖率有不同的含义。测试自动化对于自动化单元测试来说是很高的。用户验收测试一般是手动进行的，可能不适合自动化测试覆盖。**  **难以自动化的需求，不考虑自动化和自动化测试覆盖。**  **通过自动化测试实现百分之百的自动化和测试覆盖率是很难实现的，例如在可用性测试中。** | |
| 1. 自动化测试范围划分 | 选取三类理财产品，网络贷款，后台借贷审核等   * 理财产品的额度买入：系统每天都会进行大批量的基金理财数据的买卖交易。要审核买入金额、和后期收益是否合法，并且理财产品过多，手工将每个理财产品的交易额度进行测试过于麻烦。因此采用自动化来做。 * 网络贷款：系统会每天给很多人员进行贷款和发放资金，发放的资金和日期都会很大的不同，所以工作量就非常庞大了而这些操作完全由人工来做极有可能出现漏测和结果偏差的问题，因此采用自动化进方式来进行自动化测试。 * 后台借贷审核：后台系统每天都会要审核大批量的借贷资金是否符合借贷的资格，信用和规范，手动执行批量数据的审核会导致整体执行效率过低，因此采用自动化来做。 |
| 1. 人员分配 | 张三【根据前期造数据】  李四【负责后期收集自动化测试结果】  王五【编写自动化脚本】 |
| 4. 工具准备 | 1.selenium：selenium 3.11.0：负责自动化框架运行  2.PyCharm：2020.2.1 编写代码的工具 vs(轻量级开发工具)  3.selenium IDE：辅助开发工具，帮助我们开发路径比较复杂的脚本  **Selenium driver（驱动）** , **selenium grid** （分布式测试）， **selenium IDE（脚本录制）**  4.Pytest + allure，unittest单元测试框架  5.谷歌、火狐浏览器、Safari、Opera：企业标准浏览器 |
| 1. 测试数据准备 | 1. 后台借贷审核分为，大额度申请，小额申请，非法额度申请等情况还要借贷期限有分为0天，1天，正常天数，非法天数，超出规定范围外的日期天数   因为额度申请和期限组合导致的用例的数据量特别大。本次采用python + xlrd 方式循环来进行集中造数据。下面是造数据的python + xlrd代码。  Xlrd + pymysql + faker  Xxxxxxx 这个地方需要按照实际请情况调用faker的不同方法造数据，然后放在excel表格里，数据库mysql也可以可以  实际造数据代码写到下方   |  | | --- | | Import xlrd  For I in range(100):  Name = Fa.name()  Xlrd.write()  fa = Faker("zh\_CN")  print(fa.name())  print(fa.address())  print(fa.profile())  print(fa.file\_name())  for i in range(100):  f = open(file=fa.file\_name(),mode="w+") | |
| 1. 搭建服务端环境《环境确认表》（**硬件和软件**） | 服务器端环境：运行平台：Cent OS（linux免费） + Unix(收费)  硬件环境：  服务器(刀片式服务器或者机架式服务器)：  型号：IBM X3650 M5(8871I05)  cpu：E5-2603 v4  内存：120G内存  机械硬盘，固态磁盘：300T  软件环境：操作系统：Cent os 7.0（中标麒麟，银河麒麟，红旗，蓝点）  中间件：大厂 + 银行 : weblogic+ nginx + db2(人大金仓，达梦) + redis（内存数据库） + zk（服务注册中心,nacos,springcloud）  Weblogic （应用服务器）+ nginx【负载服务器】 + （IBM）db2（存储在磁盘上） + redis（内存数据库）【存储数据】 + nacos,zookeeper(分布式服务连接)【服务和服务的关联：商城和物流系统进行系统关联】 |
| **中期：** | 执行脚本进行测试 |
| 1. 脚本开发 | 如何将用例转化脚本思路：（NoSuchElementException）  先使用python开发一整套能跑的通的流程代码。然后进行增强。   1. 【数据】因为贷款审核和贷款期限的组合情况导致大量测试数据，针对这支业务，涉及到大量的数据，所以集中将数据放在excel表格中使用python进行读取。 2. 【逻辑操作】另外登陆到审核的过程的逻辑操作每次都需要重复的执行。所以将重复操作逻辑功能封装成方法方便重复调用。 3. 【用例执行】使用pytest + yaml文件进行执行用例测试 |
| 1. 调试、增强脚本 | 1. 删除selenium IDE录制时候的无效的脚本，添加没有录制进去的模块的脚本 2. 将大量重复的操作的页面逻辑业务代码封装成可复用的功能进行调用。重复使用的数据可以提取成全局变量。 3. 修改无法使用的定位方式，采用其他定位方式来代替。   Find\_element\_by\_id()过时 find\_element(By.XPATH,””)  形成：**数据读取模块 + 页面逻辑执行模块 + 用例执行模块** + 报告生成（HTMLTestRunner）和邮件发送模块(send mail) + **Jenkins(CI持续集成)执行模块 + docker（容器，可方便移植）** |
| 代码模式如何写？ | 【数据读取模块】(伪代码程序粘进来) 方便与后期协同开发，格式一旦确定，后期编写的时候统一，方便管理  def readYaml():    【页面逻辑执行模块】  def login (driver,name,pwd):  Driver.find\_element().send\_key(name)  Driver.find\_element().send\_key(pwd)  ……    【用例执行模块】  Def testLogin():  Driver = webdriver.Chrome()  Driver.get()  Driver.max()  Login() |
| **后期：** | 分析收集的结果，定位问题，然后提交bug，回归测试，编写测试总结 |
| 1. 分析结果提交bug | 1. 分析bug是自动化代码执行本身的问题还是测试出的功能的bug 2. 核对和验证bug以及提交bug到管理平台（bugfree,QC,bugzilla,禅道）。 |
| 1. 编写测试报告 | 根据测试结果对bug进行分类，编写最终测试报告。并邮件发送给各位领导。 |
| 1. 跟踪迭代和维护自动化脚本 | 随着系统版本的迭代一起进行自动化脚本的更新和维护。Gitee(互联网管理凭条),svn(公司内网平台) |

|  |  |
| --- | --- |
| 时间规划 | \_\_1.5\_\_ 周  每天的任务工作安排  1~2：前期工作的评估  3~6：编写测试脚本  7~8：分析测试报告 + 提交bug |
| 项目风险以及规避 | 1. 人员风险：   测试半截人员离职，申请加人，加班加点测试  人员水平不行，加班给他培训一下。能正常写自动化即可。   1. 项目风险：   有紧急的项目需要测试，目前项目要做断点备份。  临时加需求加功能，进行测试，开发。   1. 系统运行风险：   环境：让运维或者开发过来帮助我们排查问题。然后更新环境搭建手册。 |
| 工作的每天代码的编写情况：一天能写多少个脚本？ | 1. 一天写个50~60条脚本，涵盖3~4个模块。 |