项目研发报告

1 引言

1.1 编写目的

项目研发报告提供了对整个项目研发过程的总结性描述，具体来说包括以下几部分：

- 详细介绍“南大测试在线业务系统”的项目背景，即该项目是在什么样的环境下开展的，我们为什么要做这个项目，项目实施的社会意义是什么，项目预期达到的目标是什么，项目的发展前景如何，以及我们有哪些优势可以来完成这个项目；

- 说明项目的需求分析过程，即我们从何处获取需求，如何分析和整理需求，并逐步细化出所有的软件功能，如何全面地理解用户的各项要求，就软件功能与客户达成一致，估计软件风险和评估项目代价，并根据需求分析的结果形成一个最终的开发计划；

- 描述整个项目组的任务安排与人员分工情况，包括项目组是由哪些成员组成的，每个成员负责的是项目的哪一部分，各自的完成情况如何，以及汇总每个成员的工作量统计结果；

- 详细说明项目的管理方法，即组内是如何讨论并协调各个成员的开发进度，如何根据需求的最新变化确定下一阶段的开发任务，以及前端组和后端组之间是如何进行对接和同步的；

- 详细介绍项目开发过程中使用到的关键技术，包括项目使用的开发框架和外部基础设施，为什么这样选择，以及代码编写过程中遇到过哪些技术难题，这些问题最终是如何解决的；

1.2 项目背景

南京大学软件测试中心是具有国家资质认定的软件测试机构，可以向社会出具具有证明作用的数据和结果。测试中心的日常业务主要是接取企业客户的软件测试委托，进行相关测试并给出测试结果。长期以来，该机构一直使用基于填写word文档的方式管理整个业务流程，现在他们希望为整个业务流程搭建一个在线管理平台。

开展本项目的目的之一，就是将南大测试中心的业务变为在线的web应用形式，客户可以通过前端页面进行数据填写，并由后端服务将数据保存在专门的数据库中；同时，本项目也是以团队合作的形式进行的，因此开展本项目的另外一个目的在于，让大家能够体验团队合作的过程以及实际的软件开发工作，从需求分析到文档编写。

本项目实施的意义就在于让同学们可以感受一下真实的软件开发究竟是怎样一种情形，软件开发不只是编写代码，或者说写代码只是软件开发过程中的一小部分。除此之外，我们还需要了解到需求分析对于整个代码编写的指导作用，人的协调对于项目进度的深刻影响，外部评价标准对项目开发本身所造成的压力。项目所预期达到的目标，一方面是帮助同学们学习掌握现代web开发技术，将“南大测试在线业务系统”成功的实现出来；另一方面，也是希望能够帮助大家为以后的工作做好准备。

我们相信将传统的word文档改为现代化的web页面形式，能够促进南大测试中心的业务发展，为南大测试中心带来更多的客户资源和合作机遇。我们的优势就在于，作为南大的学生我们都有着扎实的专业基础、优秀的学习能力，可以在规定时间内完成既定的任务目标。

2 研发过程

2.1 需求分析

在线业务WEB应用平台主要是要针对具体测试项目的相关事宜进行管理。具体来说包括用户管理、客户管理、委托管理、合同管理、样品管理、报告管理、测试管理、在线审批等等功能模块。测试在线业务平台基于一系列经过需求调研后既定的流程进行开发，需要对整个项目可能进行到的流程状态进行管理，并基于此提供对应的服务和相关的权限控制。权限控制既包括不同类型人员可以进行的操作类型，也包括在不同的流程阶段可以进行的操作权限控制。在线业务平台用户包括以下几种类型：超级管理员ADMIN、客户CUSTOMER、市场部主管MARKETING_SUPERVISOR、市场部员工MARKETER、测试部主管TESTING_SUPERVISOR、测试部员工TESTER、质量部主管QA_SUPERVISOR、质量部员工QA，应对他们施加不同的操作权限控制。

待补充

2.2 项目任务

项目任务总体上可以分为两部分，一部分由前端组负责，即通过HTML、CSS及JavaScript以及衍生出来的各种技术、框架、解决方案，来实现互联网产品的用户界面交互；另一部分由后端组负责，即通过Java语言及其相关生态中被广泛使用的代码框架和第三方依赖，来实现各种数据的增删改查、业务的流程控制和权限管理。更具体的说，后端组的任务又可以分为：

1. 需求分析。
2. 代码编写
    1. 用户管理。用户可以进行注册、登录、登出，修改账号信息，进行身份认证以及权限鉴别；
    2. 委托管理。用户可以填写并提交委托，随时查看委托状态，测试中心人员可以对委托进行审核；
    3. 合同管理。委托双方可以进行合同创建、查询、评审、修改等操作，并封存合同扫描件；
    4. 样品管理。测试工作人员可在平台维护接收到的样品实体的信息、状态。
    5. 文档生成。用户可以根据已有的表单信息请求生成PDF格式的文档并下载留存。
    6. 测试项目。用户可以进行测试期间的文档的管理、测试期间的报告的管理。
3. 文档编写
    1. 需求文档。
    2. 项目研发报告。
    3. 项目测试报告。
    4. 设计说明书。

2.3 人员分工

项目组成员一共有12个人，其中前后端各有6个人：

- 前端组：毕一帆（组长），蔡鸿彬，孙思敏，肖毓哲，严思远，张宇晨
- 后端组：杨茂琛（组长），张世茂，戴显灏，姜宁，王岳，孙逸扬

具体而言，各个成员负责和完成的主要任务如下所示：

- 后端组
    - 杨茂琛：编写了文档服务的OSS的业务代码及单元测试，作为核心成员进行了2次项目重构，编写了测试微服务中近一半的单元测试，编写了测试微服务的全部dao层集成测试，作为核心成员进行项目部署、前后端联调，构建自动化工作流，编写了项目重构报告并参与其他文档的完善。
    - 张世茂：针对后端业务代码中测试服务部分的项目管理、测试文档服务、测试方案评审表服务等业务代码进行充分测试，并完善和修改测试过程中发现的业务代码中的部分问题。在开发过程中负责业务需求的调研、管理和控制，并作为产品经理和委托方及前端相关人员沟通需求问题，完成了需求文档的编写。完成了需求规格说明书的整体编写。
    - 戴显灏：完成了测试方案服务、测试报告检查表服务、委托测试工作检查表服务的代码编写，完成项目设计说明书、项目测试报告的文档全部编写工作。
    - 姜宁：完成了测试项目管理的测试报告、测试用例表、测试记录表、测试问题清单等服务的代码编写；完成了PDF文档的生成包括测试方案表（JS006）和测试报告（JS007）以及PDF的单元测试，协助完成其他表的包括初始化表项以及代码异常检查等工作。
    - 王岳：参与项目需求调研，负责与测试中心王老师沟通；完成了测试项目的projcet模块及schemeReview业务；完成了PDF文档生成任务(11/14)及测试任务；完成了样品管理服务业务及测试代码；作为核心成员进行了1次项目重构；参与项目部署及前后端联调；参与编写了需求规格说明文档。
    - 孙逸扬：完成了用户管理、委托管理、合同管理以及应用网关的功能代码和测试代码编写，与其他成员协作完成后端服务和数据库的更新部署，以及API文档、研发文档的部分编写工作。
- 前端组
    - 毕一帆：安排分工，控制进度、组织会议、同步进展，作为产品经理与后端沟通、确认需求，项目初期产出产品原型、活动图、页面图、需求文档，编写测试报告及文档展示、审核、填写跳转，不同角色不同阶段的进度展示及变化，客户审核报告，测试项目列表及人员分派，测试阶段扫描件的上传和文件下载等代码，并进行部分代码重构，编写测试脚本框架及模板并进行测试，生成测试报告，编写测试报告前端部分，与后端成员协同完成设计说明书、项目测试报告、项目研发报告，制作汇报ppt、进行日常进度汇报
    - 蔡鸿彬：早期编写表格模板，模拟后端服务器，完成委托填写、委托列表展示、分派界面、文档上传、PDF下载、样品集的填写和列表、制作首页、编写测试脚本，进行项目重构偿还技术债
    - 孙思敏：初版合同展示、合同审核，测试方案审核表填写和展示、测试报告的填写和展示，提供后端需要的api请求响应截图和文档所需界面截图
    - 肖毓哲：报价单填写和展示，合同填写、审核和展示，测试用例表，测试记录表，测试问题清单，工作检查表，报价单同意和展示，合同填写和展示。测试部分报价单填写，合同填写，报价单同意，合同填写，委托填写。
    - 严思远：用户管理功能及界面的实现，包括用户登录、用户注册、账户信息、委托及测试项目的进度查看页面、软件文档评审表和测试报告检查表、服务器部署与维护、用户手册的文档部分及展示界面、需求规格说明书的流程图部分
    - 张宇晨：需求文档、接口文档、委托审核、通用功能函数、JS006及其展示、Ant Design Pro样式库切换和测试

其中有多项任务是多成员合作完成的，基本上每个成员都负责了多个不同的任务。

2.5 数据统计

- 前端组
  - 毕一帆：16个文件，2183行代码
  - 蔡鸿彬：4543行代码
  - 孙思敏：2322行代码
  - 肖毓哲：15个文件，2774行代码
  - 严思远：3629行代码
  - 张宇晨：2059行代码
  - 共525个文件，81933行代码，除去未经修改的框架代码外总代码量为17959行，所有commit共471个。

GitHub中main分支修改记录如下：

```
Contribution stats (by author) on the 'main' branch:

     FibonaccciYan <1363650301@qq.com>:
      insertions:    285265 (32%)
      deletions:     54597  (6%)
      files:         2809   (20%)
      commits:       55 (18%)
      lines changed: 339862 (19%)
      first commit:  Fri Apr 8 00:27:16 2022 +0800
      last commit:   Fri Jul 8 13:02:46 2022 +0800

     ZFirozen <zfirozen@163.com>:
      insertions:    72177  (8%)
      deletions:     59368  (7%)
      files:         470    (3%)
      commits:       27 (9%)
      lines changed: 131545 (7%)
      first commit:  Thu Apr 7 23:35:12 2022 +0800
      last commit:   Fri Jul 8 18:58:42 2022 +0800

     Simin-Sun <1196953351@qq.com>:
      insertions:    47363  (5%)
      deletions:     34637  (4%)
      files:         263    (2%)
      commits:       14 (4%)
      lines changed: 82000  (5%)
      first commit:  Wed May 25 21:13:36 2022 +0800
      last commit:   Fri Jul 8 11:56:08 2022 +0800

     0XYZxyz0 <80372018+0XYZxyz0@users.noreply.github.com>:
      insertions:    4137   (0%)
      deletions:     3235   (0%)
      files:         42 (0%)
      commits:       30 (10%)
      lines changed: 7372   (0%)
      first commit:  Sun May 15 18:12:51 2022 +0800
      last commit:   Thu Jul 7 22:56:31 2022 +0800
```

```
junzhi-cn <80112517+junzhi-cn@users.noreply.github.com>:
 insertions:    2243   (0%)
 deletions:     1814   (0%)
 files:         39 (0%)
 commits:       9  (3%)
 lines changed: 4057   (0%)
 first commit:  Fri Apr 29 22:16:23 2022 +0800
 last commit:   Mon Jun 27 13:00:20 2022 +0800

ZFirozen <zfirozen@smail.nju.edu.cn>:
 insertions:    196    (0%)
 deletions:     13756  (2%)
 files:         47 (0%)
 commits:       3  (1%)
 lines changed: 13952  (1%)
 first commit:  Sun Apr 10 20:20:30 2022 +0800
 last commit:   Sun Apr 10 21:40:00 2022 +0800

FibonaccciYan <58576758+FibonaccciYan@users.noreply.github.com>:
 insertions:    0  (0%)
 deletions:     7  (0%)
 files:         3  (0%)
 commits:       1  (0%)
 lines changed: 7  (0%)
 first commit:  Tue Jul 5 23:32:16 2022 +0800
 last commit:   Tue Jul 5 23:32:16 2022 +0800

0XYZxyz0 <792037332@qq.com>:
 insertions:    60097  (7%)
 deletions:     214178 (24%)
 files:         2510   (17%)
 commits:       16 (5%)
 lines changed: 274275 (15%)
 first commit:  Fri Apr 8 14:05:24 2022 +0800
 last commit:   Tue Jul 5 17:12:25 2022 +0800

junzhi-cn <junzhi_cn@qq.com>:
 insertions:    392775 (43%)
 deletions:     472455 (52%)
 files:         7843   (55%)
 commits:       94 (30%)
 lines changed: 865230 (48%)
 first commit:  Fri Apr 8 21:01:56 2022 +0800
 last commit:   Fri Jul 8 23:54:50 2022 +0800

ZFirozen <80153444+ZFirozen@users.noreply.github.com>:
 insertions:    3  (0%)
 deletions:     0  (0%)
 files:         2  (0%)
 commits:       2  (1%)
 lines changed: 3  (0%)
 first commit:  Sun Mar 20 20:35:45 2022 +0800
 last commit:   Thu Apr 7 23:41:48 2022 +0800

ChinoJunko <598991406@qq.com>:
```

```
    insertions:    39872  (4%)
    deletions:     51262  (6%)
    files:         330    (2%)
    commits:       63 (20%)
    lines changed: 91134  (5%)
    first commit:  Fri Apr 8 16:13:13 2022 +0800
    last commit:   Fri Jul 8 00:34:42 2022 +0800

  total:
    insertions:    904128 (100%)
    deletions:     905309 (100%)
    files:         14358  (100%)
    commits:       314    (100%)
```

- 后端组
- - 杨茂琛：6966行增加，5293行删除，改动549个文件，165次提交
  - 孙逸扬：22174行增加，17407行删除，改动1113个文件，53次提交
  - 戴显灏：4111行增加，785行删除，改动213个文件，43次提交
  - 姜宁：6920行增加，1396行删除，改动232个文件，57次提交
  - 王岳：13175行增加，4401行删除，改动551个文件，82次提交
  - 张世茂：3342行增加，903行删除，改动86个文件，43次提交

```
Contribution stats (by author) on dev branch:

        Johnny J <1149224994@qq.com>:
         insertions:    6920   (12%)
         deletions:     1396   (5%)
         files:         232    (8%)
         commits:       57     (12%)
         lines changed: 8316   (10%)
         first commit:  Wed Apr 20 21:52:14 2022 +0800
         last commit:   Sat Jul 2 14:24:33 2022 +0800

        syhien <32589592+syhien@users.noreply.github.com>:
         insertions:    170    (0%)
         deletions:     39     (0%)
         files:         20     (1%)
         commits:       20     (4%)
         lines changed: 209    (0%)
         first commit:  Fri Jun 3 17:33:03 2022 +0800
         last commit:   Fri Jul 1 20:17:07 2022 +0800

        stu <wangyue1@smail.nju.edu.cn>:
         insertions:    13175  (23%)
         deletions:     4401   (15%)
         files:         551    (20%)
         commits:       82     (18%)
         lines changed: 17576  (20%)
         first commit:  Wed May 4 12:37:09 2022 +0800
         last commit:   Tue Jul 5 00:32:24 2022 +0800

        Xianhao-Dai <438081108@qq.com>:
```

```
 insertions:    4111   (7%)
 deletions:     785    (3%)
 files:         213    (8%)
 commits:       43     (9%)
 lines changed: 4896   (6%)
 first commit:  Tue Apr 19 17:50:15 2022 +0800
 last commit:   Sat Jul 2 13:26:12 2022 +0800

syhien <syhien@outlook.at>:
 insertions:    6966   (12%)
 deletions:     5293   (18%)
 files:         549    (20%)
 commits:       165    (35%)
 lines changed: 12259  (14%)
 first commit:  Mon Apr 18 22:04:29 2022 +0800
 last commit:   Tue Jul 5 17:35:01 2022 +0800

Yiyang Sun <59226324+YiyangSunn@users.noreply.github.com>:
 insertions:    234    (0%)
 deletions:     3      (0%)
 files:         6      (0%)
 commits:       4      (1%)
 lines changed: 237    (0%)
 first commit:  Fri Apr 15 13:56:27 2022 +0000
 last commit:   Fri Jul 8 20:08:55 2022 +0800

YiyangSunn <1196168404@qq.com>:
 insertions:    22174  (39%)
 deletions:     17407  (58%)
 files:         1113   (40%)
 commits:       53     (11%)
 lines changed: 39581  (45%)
 first commit:  Sat Apr 16 00:29:15 2022 +0000
 last commit:   Fri Jul 8 20:09:30 2022 +0800

zhangshimao <1345168596@qq.com>:
 insertions:    3342   (6%)
 deletions:     903    (3%)
 files:         86     (3%)
 commits:       43     (9%)
 lines changed: 4245   (5%)
 first commit:  Thu May 26 12:33:27 2022 +0800
 last commit:   Thu Jun 30 00:12:44 2022 +0800

total:
 insertions:    57092  (100%)
 deletions:     30227  (100%)
 files:         2770   (100%)
 commits:       467    (100%)
```

2.6 项目管理

前端组采用每周例会+不定期快速会议来确保每周的信息同步和出现问题后的快速反应。在项目中后期，为了确保与后端组对接效率，前后端组在每周都会进行1-2次联合会议，确定对接功能细节。在平时，前端组使用qq群来进行信息的传递，同时也可以解决小问题。对于部分方便线下见面的组员，有时也会采取线下联合开发的方式，利用线下交流帮助修复bug，实时沟通，提高开发效率。

在代码管理方面，前端组使用GitHub作为代码仓库，每个人在自己的分支下进行开发，并统一合并到主分支。

3 关键技术

3.1 微服务架构

后端代码采用微服务架构进行开发。微服务是一种软件架构风格，它是以专注于单一责任与功能的小型功能区块为基础，利用模块化的方式组合出复杂的大型应用程序，各功能区块使用与语言无关的API集相互通信。微服务运用了以业务功能为主的设计概念，应用程序在设计时就以业务功能或流程设计先行分割，将各个业务功能独立实现成一个能自主执行的个体服务，然后再利用相同的协议将所有应用程序需要的服务都组合起来，形成一个整体。

在本项目中，每个管理模块都是一个单独的微服务应用，各个服务之间通过HTTP协议进行通信，通过外部数据库来实现数据同步和状态共享。采用微服务架构的优点就在于，当需要针对特定业务功能进行扩展时，只需要对该业务功能的服务进行扩展就好，不需要整个应用程序都扩展；当某个服务模块出现问题不能正常工作时，其它不相关的服务仍然可以正常工作。同时，微服务架构因为使用更小、更独立的划分单元，从而可以方便的进行水平向扩展与集群配置，每个服务可以独立部署并且快速启动，可以由专门的团队负责专门的服务。

3.2 Spring Framework

后端代码主要基于Spring框架进行开发，Spring Framework是Java语言最受欢迎、使用最广泛的代码框架之一。它是一个开源的全栈应用程序框架和控制反转容器实现，通过依赖注入把创建对象的权利交给框架，由框架来管理对象的存储与生命周期；通过面向切面编程将那些与具体业务无关，但被不同模块同时使用的逻辑抽离并封装起来，降低了系统的冗余度与耦合度。Spring为诸多第三方依赖和基础设施提供了整合，也为微服务架构提供了开箱即用的实现。具体来说，我们的项目使用到了以下的Spring模块：

- Spring Boot：使用基于注解和约定的自动配置，使程序员可以摆脱繁琐冗长的XML配置文件，大大简化了Spring应用的搭建和开发过程；通过集成大量的第三方工具使得依赖包的版本冲突得到很好的解决；
- Spring Cloud：极大简化了基于微服务的分布式系统开发，包括整合Eureka以提供服务注册和发现，提供基于响应式编程的Gateway，以及用于负载均衡的LoadBlancer来实现服务间调用等等；
- Spring Data：为数据访问提供熟悉且一致的基于Spring的编程模型，同时仍然保留底层数据存储的特殊特性；提供了对常见数据库如MySQL、Redis、MongoDB的整合，使得访问数据库变得简单易用；
- Spring Session：专注于解决分布式或集群配置场景中的会话管理问题，可以简单快速的集成第三方缓存数据库，从而实现会话数据共享；
- Spring Test：是用于测试Spring应用的程序框架，它为Spring的各个模块提供了测试方法，可以方便的整合JUnit5以实现单元测试，整合Jacoco以实现代码覆盖率统计，同时也可用于集成测试。

除此之外还有其它的一些Spring提供的功能，这里不再一一列举。基于Spring框架进行开发，深刻体现了软件工程中关于代码复用的哲学，即如何利用已有的第三方支持，更快更好的完成项目开发。

3.3 数据库相关

由于后端应用主要就是为前端提供数据服务，因此数据库的选择和使用非常重要。将数据放在专门的数据库中，一方面有利于数据完整性和安全性；另一方面，由于我们使用的是微服务架构，要想实现数据共享必须将数据放在外部的数据库中。选择正确的数据库不仅有利于代码编写，还能提升应用的整体性能。

3.3.1 MySQL

MySQL是最流行的关系型数据库管理系统之一，由于其性能高、成本低、可靠性好，被广泛应用在各种web应用中。MySQL使用C和C++语言编写，并使用了多种编译器进行测试，具有良好的可移植性。MySQL有可以免费使用的社区版本，并且它为多种编程语言提供了API，其中就包括Java语言的MySQL Connector。

MySQL适合存储大量的关系型数据，能够支持并发访问和事务管理。本项目中使用MySQL保存用户的基本信息，包括用户名、密码和用户身份等等。因为现在容器化技术比较流行，MySQL也提供了Docker容器版本，所以本项目中使用的MySQL其实是Docker容器，跟真正的MySQL Server比它的好处就是不需要改配置文件就能用。

3.3.2 Redis

Redis是最流行的非关系型数据库管理系统之一，它使用C语言编写，是一个高性能的键值对存储数据库。本项目中使用redis主要是为了管理会话，Spring Session提供了与Redis相关的集成，用户登录后的会话数据都保存在外部的Redis数据库中，这样就实现了会话状态在集群间共享。网关服务会向用户服务请求用户的身份信息，以实现后续的权限鉴别；而后者会从Redis数据库中取出保存在会话里的用户数据，这样就可以避免每次都访问MySQL带来的磁盘I/O开销。另外，如果要将用户服务部署成集群模式，多个用户服务实例之间通过Redis共享，将不会丢失会话数据。本项目使用的其实也是Redis的Docker镜像，不过修改了配置文件以加上密码。

3.3.3 MongoDB

MongoDB也是最流行的非关系型数据库管理系统之一，与Redis不同的是，前者主要用于键值对存取；而它主要用于文档对象存取。MongoDB和Redis都提供了Java语言的驱动版本，Spring框架也为这两个数据库提供了整合方案。MongoDB主要将数据存储在内存中，不过它会利用底层操作系统的内存映射机制，每隔一段时间把数据刷到磁盘上以实现持久化。

本项目使用MongoDB来存储各种表格数据，包括委托表、合同表、测试方案表等等。这些表格都有很多字段，并且字段都是不定长的字符串，使用MySQL这样的关系型数据库存储会很不方便。而MongoDB由于采用基于JSON对象的存储形式，可以很方便的存取这些表格。另外，由于MongoDB本身自带缓存，因此也不需要为它单独准备一个缓存方案，非常的好用。与前面两个类似，本项目使用的MongoDB也是Docker镜像。

3.3.4 MyBatis

MyBatis是一个半ORM框架，它支持自定义SQL，并且免除了几乎所有的JDBC代码以及设置参数和获取结果集的工作，是一个广泛使用的持久层解决方案。MyBatis可以通过简单的XML或注解来配置和映射原始类型、接口和 Java POJO（普通老式 Java 对象）为数据库中的记录。与Spring Data JPA默认使用的Hibernate框架不同的是，前者是一个全自动的ORM框架，从生成SQL到获取查询结果都是自动完成的；而MyBatis保留了由程序员编写SQL的步骤，这使得优化SQL查询更加方便，这也是它被称为半ORM的原因。本项目主要使用MyBatis来访问MySQL数据库，而使用它的好处就在于我们只需要编写SQL语句就行了。

3.3.5 Object storage

对象存储（也称为基于对象的存储）是一种计算机数据存储，它将数据作为对象进行管理，而不是像文件系统那样将数据作为文件层次结构进行管理，以及块存储那样将数据作为扇区和轨道内的块进行管理的其他存储架构。每个对象通常包括数据本身、数量不等的元数据和一个全局唯一的标识符。对象存储试图实现其他存储架构所不具备的能力，如可由应用程序直接编程的接口，可跨越多个物理硬件实例的命名空间，以及数据管理功能，如数据复制和对象级颗粒度的数据分发。

绝大多数对象存储的实现都是云存储。基于云存储，本业务平台的二进制文件的分发可以独立于后端服务器的运行；提升在线率和可用性的同时，优化了二进制文件分发的用户体验。

3.4 构建工具

构建工具是用来管理项目的各个模块，以及它们所使用的第三方依赖的。它为整个项目提供了从源码编译到运行测试，再到打包部署的完整流水线支持。任何一个适合大规模项目开发的语言，都一定会有一个与之对应的构建工具，它管理着项目中用到的第三方依赖的版本，以便在不同成员的不同开发环境下都能顺利编译该项目。

3.4.1 Maven

Maven是Java语言使用最广泛，也是最可靠的构建工具之一，它由Apache基金会编写和维护。基于项目对象模型（POM）概念，Maven可以从一条中心信息管理项目的构建、报告和文档。Maven有一个中央仓库，许多第三方应用都会在这个仓库中发布自己的各个版本，包括Spring框架、MySQL驱动以及MyBatis等等。

Maven使用基于XML的配置文件，并且支持各种各样的第三方插件。比如项目中使用Spring官方提供的Maven插件spring-boot-maven-plugin来打包部署Spring应用，这个插件会把所有用到的第三方依赖都打包进去，这样打出来的JAR包是可以直接运行的；同时还使用了maven-failsafe-plugin，这个插件提供了在Maven项目中运行集成测试的方法。

3.5 PDF书写工具

便携式文档格式（英语：Portable Document Format）是一种用独立于应用程序、硬件、操作系统的方式呈现文档的文件格式。每个PDF文件包含固定布局的平面文档的完整描述，包括文本、字形、图形及其他需要显示的信息。本业务平台需要根据运行时的信息生成稳定排版、格式良好的PDF文档以用于测试中心内部归档、客户审阅。

3.5.1 iText

iText是一个用于在Java和.NET中创建和操作PDF文件的开放源代码库。iText提供对高级PDF功能的支持，如基于PKI的签名、40位和128位加密、颜色校正、标签PDF、PDF表格（AcroForms）、PDF/X、通过ICC配置文件和条形码的颜色管理，并被一些产品和服务使用，包括Eclipse BIRT、Jasper Reports、JBoss Seam、Windward Reports和pdftk。

iText有着不俗的运行效率，后端可以稳定地保证在可接受的较短时间内生成PDF完毕并准备分发。本业务平台通过iText提供的API接口从空白页面开始，从头按着指定格式和内容生成PDF文档。通过对字体、排版等样式的指定，本业务平台可以生成格式文档、可用于正式商业环境的PDF文档。

3.6 持续集成

持续集成是一种 DevOps 软件开发实践。采用持续集成时，开发人员会定期将代码变更合并到一个中央存储库中，之后系统会自动运行构建和测试操作。持续集成通常是指软件发布流程的构建或集成阶段，需要用到自动化组件（例如 CI 或构建服务）和文化组件（例如学习频繁地集成）。

3.6.1 Github Actions

GitHub Actions 让团队轻松实现软件工作流程的自动化。从GitHub上直接构建、测试和部署团队的代码。让代码审查、分支管理和问题处理以简单、明晰、高效的方式进行。通过GitHub事件（如push、pull request或release）来启动工作流程。使用由社区建立和维护的服务进行组合并配置行动。

Github Actions 无需自有长期开放、性能足够的服务器。Github Actions 在 Microsoft 提供的虚拟机上全新地装载指定的运行环境，开箱即用、用后即销毁无需担心残留和隐私问题。得益于全新的虚拟机环境，Github Actions 可以并行多个自动化任务。本团队利用 Github Actions 进行：1. 自动化构建；2. 自动化测试；3. 自动化覆盖度测试（提供可下载的覆盖度报告）；4. 代码扫描警报。

3.7 React – 框架

React是用于构建用户界面的JavaScript库，主要用于构建UI。React有以下特点：

1. 声明式设计：React 使创建交互式 UI 变得轻而易举。为你应用的每一个状态设计简洁的视图，当数据变动时 React 能高效更新并渲染合适的组件。
2. 组件化: 构建管理自身状态的封装组件，然后对其组合以构成复杂的 UI。
3. 高效：React 通过对 DOM 的模拟，最大限度地减少与DOM的交互。
4. 灵活：无论你现在使用什么技术栈，在无需重写现有代码的前提下，通过引入 React 来开发新功能。

3.8 Ant Design & Ant Design Pro – 组件

antd 是基于 Ant Design 设计体系的 React UI 组件库，主要用于研发企业级中后台产品。antd 提炼自企业级中后台产品的交互语言和视觉风格，是开箱即用的高质量 React 组件。antd 使用 TypeScript 开发，提供完整的类型定义文件，使用全链路开发和设计工具体系。

Ant design Pro开箱即用的中台前端/设计解决方案，为 Web 应用提供了丰富的基础 UI 组件, 是一套企业级 UI 设计语言和 React 组件库

3.9 JSONSERVER – 模拟服务器

json-server是一款基于Node.js的服务器，为前端开发人员可以提供一个高仿真的RESTFul后台服务（数据原型），用于模拟服务器。在项目早期，还没有真正后端的时候，前端组使用JsonServer模拟了后端服务器，包括数据库、服务层和部分业务层的功能，并使用模拟的服务器去测试我们的前端界面。

3.10 PostMAN – 调试

Postman是一个用于构建和使用API的API平台。Postman简化了API生命周期的每个步骤，并优化了协作，因此我们可以更快地创建更好的API。前端组为了调试模拟服务器，使用PostMan在调试时发送请求。

3.11 Ant design landing – 首页设计

Ant Design Landing 是针对产品首页的解决方案，秉承 Ant Design 的设计价值观，延用 Ant Design 的设计原则，可以快速搭建出想要的首页，进一步提升工作效率。Ant-design-landing是遵循antd设计理念的首页编辑器，在antdpro的脚手架中十分易用，前端组用它生成模板，并在此基础上进行修改，生成首页。

3.12 Umi/dumi - 搭建用户手册展示页面

umi 是一个可插拔的企业级 react 应用框架。它具有插件化、开箱即用、约定式路由等特点。

主要特性：

- 可扩展，Umi 实现了完整的生命周期，并使其插件化，Umi 内部功能也全由插件完成。此外还支持插件和插件集，以满足功能和垂直域的分层需求。
- 开箱即用，Umi 内置了路由、构建、部署、测试等，仅需一个依赖即可上手开发。并且还提供针对 React 的集成插件集，内涵丰富的功能，可满足日常 80% 的开发需求。
- 大量自研，包含微前端、组件打包、文档工具、请求库、hooks 库、数据流等，满足日常项目的周边需求。
- 完备路由，同时支持配置式路由和约定式路由，同时保持功能的完备性，比如动态路由、嵌套路由、权限路由等等。

dumi是基于 Umi 打造、为组件开发场景而生的文档工具，可以用来写文档、官网和组件库 Demo 。前端组使用dumi进行用户手册展示界面的搭建。

3.13 Nginx - 搭建服务器

Nginx (engine x) 是一个高性能的HTTP和反向代理web服务器，同时也提供了IMAP/POP3/SMTP服务。它的稳定性好，功能集丰富，配置文件简单，系统资源消耗低。它是一款轻量级的 Web 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器，在BSD-like 协议下发行，占有内存少，并发能力强。

前端组使用Nginx来搭建服务器并运行。

3.14 Pytest – 测试

pytest是一个非常成熟的全功能的Python测试框架，主要有以下几个特点:简单灵活，容易上手，支持参数化，能够支持简单的单元测试和复杂的功能测试，可以用来做selenium/appnium等自动化测试、接口自动化测试。pytest具有很多第三方插件，并且可以自定义扩展。pytest支持测试用例的skip和fail处理，可以很好的和jenkins集成。

前端组使用pytest进行前端页面的功能测试。

3.15 Selenium – 自动化工具

Selenium是一个用于Web应用程序测试的工具，测试直接运行在浏览器中，就像真正的用户在操作一样。Selenium框架底层使用JavaScript模拟真实用户对浏览器进行操作。测试脚本执行时，浏览器自动按照脚本代码做出点击，输入，打开，验证等操作，就像真实用户所做的一样，从终端用户的角度测试应用程序。它使浏览器兼容性测试自动化成为可能，尽管在不同的浏览器上依然有细微的差别。它的使用简单，可使用Java，Python等多种语言编写用例脚本。

前端组使用selenium模拟用户操作，进行前端页面测试。