START! 软件项目总结报告

1. 引言

1.1. 编写目的

本软件项目总结报告文档的编写目的是对 Study Together And Review Together! 在线学习与复习单词系统（以下简称 START!）的整个项目工作进行总结和回顾。文档将展示项目的最终结果，对开发工作进行评价，并总结经验和教训。本文档用于开发团队总结项目情况，为之后的工作积累经验。

1.2. 适用范围

本文档适用的软件：START!

与该软件相关的特性、子系统、模型、代码等均符合本文档中的内容。

1.3. 定义

本文件中涉及的术语定义在项目词汇表中给出。

1.4. 参考资料

《面向对象软件工程：使用 UML、模式与 Java》（第 3 版），清华大学出版社， 2011。

《面向对象软件工程实践指南》，上海交通大学出版社，2016。

1.5. 概述

本文档包括实际开发结果、开发工作评价及经验与教训三部分。实际开发结果总结了项目的产品、费用和人员等情况。开发工作评价从多个方面回顾开发工作。经验与教训是整个团体最后的反思和体会。本文件的各部分内容联系紧密，回顾了之前的工作，进行了全面的总结。各部分互为补充和对照，共同呈现本项目的总体情况。

2. 实际开发效果

2.1. 产品

2.1.1. 程序系统

2.1.1.1. 客户端程序层级关系（见表 1）

表 1 程序层级关系表

包名	程序名	程序大小（单位：KB）
creat-react-app	joinRoom.js	4
	login.js	3
	main.js	8
	personal.js	4
	register.js	8
	reviewRoom.js	24
	reviewSummary.js	8
	reviewWait.js	16
	searchWord.js	4
	studyRoom.js	16
	studySummary.js	4
react-bootstrap	init.js	4
	App.js	4
	App.test.js	0.3
	App.todo.js	8
	index.js	0.3
	setupTest.js	0.2
	Footer.js	0.3
	Logo.js	0.2
	Nav.js	4
	Nav2.js	0.5
	Modal.js	2

2.1.1.2. 服务器端程序（见表 2）

表 2 服务器端程序列表

包名	程序名	程序大小（单位：KB）
backend	asgi.py	1
	settings.py	4
	urls.py	3
	wsgi,py	1
	__init__.py	1
start	admin.py	1
	apps.py	1
	concreteclass.py	13
	models.py	2
	roomControl.py	9
	serializers.py	1
	UseLoginControl.py	3
	views.py	26
	wordControl.py	3
	__init__.py	0

2.1.2. 程序系统版本（见表 3）

表 3 程序版本列表

程序系统名	程序版本号	描述	修订日期
start（客户端程序）	v1.0	功能实现版本	2020-6-15
	v1.1	进行了前端界面优化	2020-6-17
startServer（服务器端程序）	v1.0	功能实现版本	2020-6-15

2.1.3. 文件描述

2.1.4. 数据库

2.2. 主要功能和性能

主要功能和性能列表如表 4 和表 5 所示。

表 4 主要功能列表

主要功能	开发目标	备注
注册	达到	
登录	达到	
查看个人信息	达到	
搜索单词	达到	
学习单词	达到	
复习单词	达到	
聊天	未完全达到	聊天框需要发一句话“激活”才能看到聊天内容
创建房间	达到	
加入房间	达到	
退出房间	达到	
查看复习结果	达到	

表 5 主要性能列表

性能需求	开发目标	备注
响应时间需求	达到	
吞吐量需求	超过预期	成功使用 websocket 代替轮询,极大降低服务器压力
容量需求	达到	
降级模式	达到	
资源需求	未完全达到	

2.3. 基本流程

里程碑事件	计划完成日期	实际完成日期	进度偏差
需求定义文档完成	2020-4-20	2020-4-20	按时完成
软件架构设计文档完成	2020-5-20	2020-4-25	提前 25 天

模块开发完成	2020-6-10	2020-6-13	延迟 3 天
系统集成完成	2020-6-15	2020-6-15	按时完成
系统测试	2020-6-17	2020-6-16	提前 1 天
项目全部结束	2020-6-19	2020-6-17	提前 2 天

2.4. 费用

由于本项目作为实习项目依托学生团队开发，因此目前暂无实际的支出。

3. 开发工作评价

3.1. 对生产效率的评价

3.1.1. 系统开发历时约 3 个月。

3.1.2. 开发反复性比较多。

3.1.3. 人员工作分配不平均，效率差异较大。

3.1.4. 程序平均生产效率，每人每月生产约 500 行。

3.1.5. 文件平均生产效率，每人每月生产约 4000 多字.

原计划：每人每月生产约 1000 行代码、 3000 字左右。

3.2. 对产品质量的评价

约每 150 条指令会出现一个 BUG，错误发生率在估计范围内。

3.3. 对技术方法的评价

3.3.1. 使用了 Django 搭建服务器端 web 框架，方便程序员快速地投入操作逻辑的开发中，同时收益于 Django 的 MVC 模式,与数据库的交互也十分方便。

3.3.2. 采用了前后端分离模式，人员分成两组分别进行开发，进度互不干扰，方便进行项目管理。

3.3.3. 采用了 websocket 建立客户端与服务器之间的长连接，服务器可以主动向客户端发送数据实现数据的实时更新，相比轮询具有高得多的效率。

3.4. 出错原因的分析

3.4.1. 使用 json 对象在客户端与服务器之间传输数据时默认采用 asc 码编码，使用 utf-8 会导致编码不正确，解决办法是在 json.dumps 时加入参数 ensure_asc=False。

3.4.2. 本来将一些初始化的函数放在项目的启动文件__init__.py 中，然而 Django 在启动时会启用两个线程运行，一个用于监视线程情况另一个为工作线程，初始化函数会被运行两遍，解决办法是将初始化函数放在视图 views.py 中，该文件在 Django 启动时只运行一次。

3.4.3. 为服务器申请了 SSL 安全证书后,原有的 ws 链接因为不安全无法使用，解决方案是改成 wss 链接并使用 nginx 进行反向代理，这样不用修改服务端接口也可以正常运行。

4. 经验与教训

4.1. 经验

4.1.1. 计划方面：从实际开发进度的对比中可以看到，实际开发进度比起计划有较大的延迟。由于是首次按照系统的流程来开发一个软件，经验不足是造成这种情况的主要因素。在计划阶段，对实际开发过程中可能遇到的各种问题考虑不足，给各阶段预留的时间较少。在实际开发过程也遇到了一些事先没有预料到的问题，加上开发组员时间安排等因素，造成后期工作量激增。

4.1.2. 需求方面：项目的需求调查一定要做到位，落实到具体使用者的需求，完全考虑产品 使用者可能会有的需求，需求没有最细，只有更细，这样才能在后续的开发中符合客户的需求， 同时在实施过程中，以需求作为准则，进行更好的分析和开发。

4.1.3. 设计方面：项目的设计中使用了 UML（统一建模语言），对项目的开发起到了很好的指导作用。对于项目开发人员，在系统的分析和设计阶段，尽可能地让其参与，保证开发目的明确，避免出现开发产品和预先设计脱节的情况，为设计一流的系统提供设计保障。

4.1.4. 技术方面：尽量根据预先设定的文档来进行开发，避免脱离初衷。项目开发遇到问 题时应该尽早提出，避免拖延，想尽一切办法解决，否则会拖延整个项目进度，对于每一个 BUG 不应该抱有侥幸心理。项目管理上应保持严格的态度。

4.2. 教训

对 Git 使用不熟悉，导致前期版本控制与分支管理出现问题，导致有时代码被覆盖；没有合理分配任务，导致部分成员必需等到其他成员完成工作后才能开始工作，造成时间浪费而导致后期工作量暴增。

4.3. 收获

这次软件开发过程提供的经验是宝贵的，使我们对一个软件从开发计划、需求获取，到分析、设计、构造，再到最终的测试、交付的全过程有了亲身的体会，也有了一定的认识和理解。虽然遇到了许多困难，也仍存遗憾，但本项目是对我们极大的磨练，也让我们更有信心面对将来更复杂的挑战。在以后的软件开发过程中，

我们对整体过程的把握一定会有所提升，考虑问题也会更加完善和周到。 由衷感谢曹老师一学期来的指导与帮助！