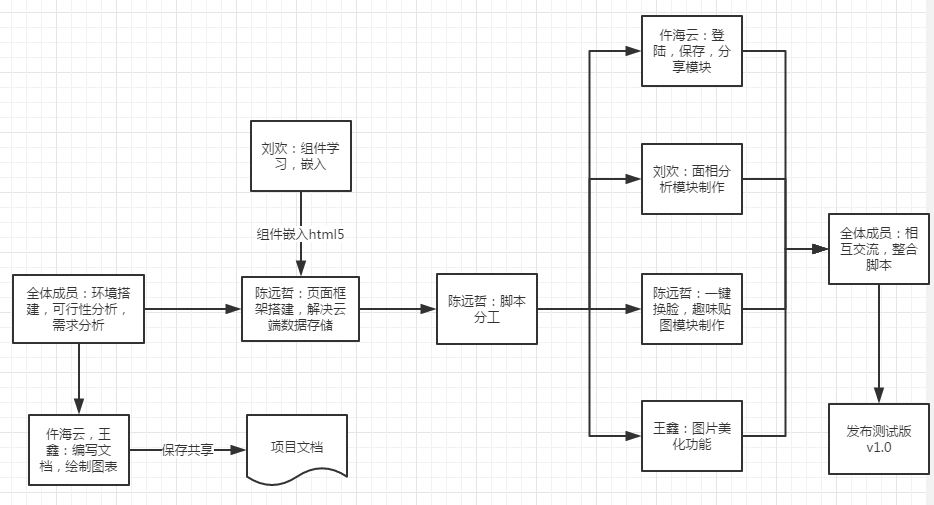
**软件工程过程**

1. **团队健康状态**

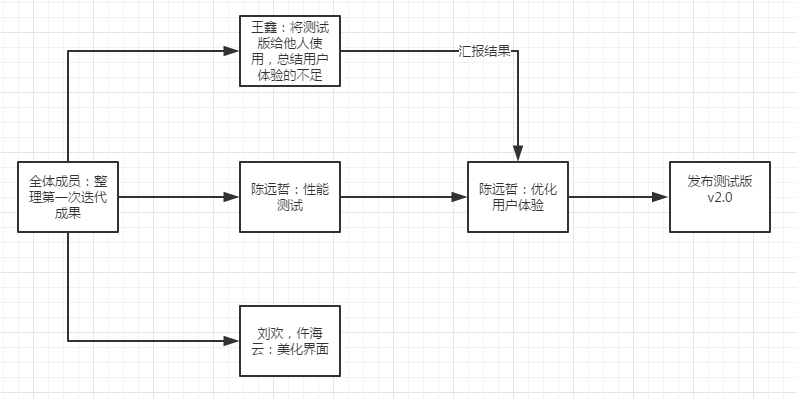
* **团队人员角色分配及职责履行情况说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **周次** | **迭代内容** | **成果版本** | **性能评估** |
| **2~10周** | 页面框架搭建，实现各个模块的基础功能 | e键换脸测试版v1.0 | 能够使用，用户能够获取到想要的东西 |
| **11~14周** | 优化用户体验，优化系统外观 | e键换脸测试版v2.0 | 用户能够轻松上手，界面良好 |
| **15~16周** | 修复bug，继续优化 | e键换脸正式版v1.0 | 系统稳定 |

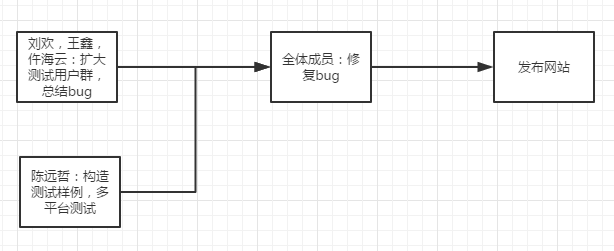
第一次迭代角色分配：



第二次迭代角色分配：



第三次迭代角色分配：



* **团队活动的组织与决策产生的过程说明**
* 活动一：小组四人对于要开发一个什么样的软件开展了研讨会

|  |  |
| --- | --- |
| 时间 | 第一周的周末 |
| 过程 | 在讨论中我们分别对生活中的一些需求展开了头脑风暴，最后再结合一些现成的例子，筛选出了一系列的可选题目。包括了（基于web的换脸工具，虚实结合的地图游戏，数学计算器）。 |
| 决策 | 由于虚实结合的地图游戏由于同时拥有社交性和游戏性，如果采用游戏引擎开发则社交方面的开发会相对比较困难，如果采用原生安卓开发则缺少游戏的趣味性，再加上现成的项目很多，因此我们放弃了这个题目。 关于数学计算器，当时我们想开发一款手写公式获取计算结果的app，但是技术水平有限，目前没有很好的解决方案，因此暂时放弃了。 因此我们最后选择开发基于web的在线换脸工具，其有以下优势：1、当前html5大热，其canvas（画布）标签能够很好的支持像素操作。2、目前市面上没有一款基于web的换脸工具，我们目前是唯一一个。3、基于web使得该工具无需下载，即可使用 4 兼容各种操作系统，兼容手机端和PC端 |

* 活动二：小组四人对于基于web的换脸工具的实用性进行调研

|  |  |
| --- | --- |
| 时间 | 第二周 |
| 过程 | 我们对舍友，同学，以及一些非IT专业的学生，以及一些成年人进行了采访。首先询问他们平时有没有这方面的需求（生活中是否经常想要恶搞图片），然后对他们希望有的功能进行了归纳总结，最后还观察他们是否期待我们的项目。 |
| 决策 | 根据大家反馈的结果，大家大多数都对我们的这个项目有着兴趣，表示想要使用。因此我们的项目迎合了大众的需求，保证了项目不是一个纯粹的作业，而是能被大众所使用的网站。因此我们项目就此定下了。 |

* 活动三：小组四人对项目的架构以及分工进行了商讨

|  |  |
| --- | --- |
| 时间 | 第二周周末 |
| 过程 | 我们分析了各种架构的可行性，分析了我们可能会使用到的各种组件。我们还对功能进行了定义和划分，根据功能来确定网站的框架。根据网站的框架，我们对每个人的开发部分进行了划分。 |
| 决策 | 最后根据每个人的优势，我们完成了分工。架构上我们采用了非传统型的B/S架构，即在传统B/S架构的基础上将瘦客户端改为胖客户端，将后台从传统的数据库改为云数据库。网站功能我们也划分为：一键换脸，趣味贴图，颜值测试，图片美化，保存与分析 5个部分。网站的UI模板我们使用了Bootstrap，人脸识别插件使用了face++，云数据库使用了bmob，项目管理工具使用了git |

* 活动四：小组四人进行项目的迭代开发

|  |  |
| --- | --- |
| 时间 | 第三周至今 |
| 过程 | 日常的开发与商讨，对不合理的需求及时修改。每天的开发记录开发日志，将测试后可运行版本上传至github。日常的对BUG 进行修复 |
|  |  |

* **项目、团队现存的主要风险及应对措施说明**

|  |  |
| --- | --- |
| 风险 | 应对措施 |
| 多平台下，多浏览器内核下的兼容性问题 | 执行脚本前先对浏览器进行识别，根据不同的浏览器用不同的程序来实现功能 |
| 前台与云服务器直传数据的稳定性 | 云端的稳定性我们无法估计，但是我们能给用户足够多的提示，提示用户稍后上传图片 |
| 工作量大于估算数 | 尽可能考虑时间成本代价，尽可能按原计划执行 |
| 团队中每个人开发的组件无法整合 | 持续集成，每天都进行一次整合 |
| 项目对浏览器的兼容性要求过于苛刻，导致许多用户无法使用我们的项目 | 后期对兼容性方面进行代码改进 |

* **团队文化的建设与贡献情况说明**

关于团队文化的建设，我们始终贯彻着“每个人都有自己的价值”这一理念，在日常开发中激励着组员们。

我们有着优秀的团队指导老师，在我们的开发过程中给我们建议和前进的方向，以及技术指导。

我们有着先锋型的团队队长，在项目开发中往往会身体力行，试图通过榜样的力量灌输给团队成员同样的品质。

我们的团队充满活力，可以从组员们的创造性、工作热情、和谐的团队氛围中体现出来

我们的团队采用比较灵活的管理方式，不要求每个人每天必须完成什么，只要求每个人每天总结自己今天干了什么，开发成果验收的时间为每一次迭代的结束，给了组员合理分配工作时间的权利

关于团队情绪的管理，我们的项目开发也经历了低潮期。但是我们不灰心不放弃，每当遇到困难，我们的团队都坚信，只要需求合理，就一定有实现的途径，因此最后我们总能客服困难。

至于贡献情况，因为我们始终贯彻着“每个人都有自己的价值”这一理念，所以我们每个人都对团队文化的建设贡献了属于自己的那一部分。

1. **软件工程过程的实施如何？**

* **在项目实践中应用了哪些RUP的最佳实践**

1 迭代的开发软件

我们的开发过程一共分为了3次迭代，第一次迭代主要为框架搭建以及基本主要功能的实现，第二次迭代继续开发新的功能和优化系统，第三次迭代主要就是优化系统，修复BUG。迭代的软件开发降低了我们项目开发的风险，加快了与用户反馈，保证最终产品不出现大偏移。

2 需求管理

在软件需求开发过程中我们使用用例和场景来捕获功能性需求。同时我们使用了用户故事来描述功能性需求和非功能性需求。我们将这些内容整理后，通过对用例，场景，用户故事的分析来推动后续软件的开发设计和测试活动。

3 使用基于构件的体系结构

我们的项目开发过程就是基于构建来进行开发的。我们在javascript开发中采用了“构件复用”的思想，将脚本分为“逻辑控制程序”和“可调用构建库”两个部分。逻辑控制程序指的就是一些自动执行的脚本代码以及一些触发事件的代码，比如界面初始化，预加载等等。而可调用构建库指的是我们将一些可能多次使用的代码段封装成构建，统一整合到若干个脚本中，供逻辑控制程序调用。比如操作的撤销，重做，一些按钮的UI行为，坐标映射等。

4 可视化软件建模

我们使用了UML建模语言，通过各种UML图来捕获体系结构和构件的架构和行为。通过可视化软件建模，我们确保构件模块一致于代码，保持设计和实现的一致性，促进明确的沟通。

5 验证软件质量

我们小组主要通过网站访问性能测试，兼容性测试，白盒测试，用户开放性测试来进行软件质量的验证。通过网站访问性能测试，我们测试了我们网站的访问性能，加载速度，并发量等数据。兼容性测试主要是在开发过程中进行的，确保网站在不同设备下的正常运行，尽早发现问题。白盒测试主要实在迭代结束后进行的，来查找逻辑功能上的缺陷。用户开发性测试是为了能发现一些我们小组发现不到的错误。

6 控制软件变更

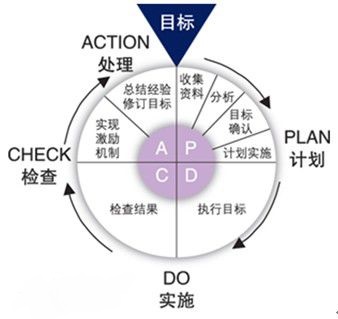
由于我们是迭代开发的，某种程度上我们降低了开发风险。并且我们在迭代过程中尽早和用户进行交互和反馈，通过同学老师们的一些建议，在开发过程中尽早变更需求，制定相应的变更处理和应对计划，减少软件变更的成本，也减少项目失败的风险。

* **在项目实践中对敏捷过程产生的具有感悟**

敏捷开发以用户的需求进化为核心，采用迭代、循序渐进的方法进行软件开发。在敏捷开发中，软件项目在构建初期被切分成多个子项目，各个子项目的成果都经过测试，具备可视、可集成和可运行使用的特征。换言之，就是把一个大项目分为多个相互联系，但也可独立运行的小项目，并分别完成，在此过程中软件一直处于可使用状态。

RUP具有很多长处：提高了团队生产力，在迭代的开发过程、需求管理、基于组件的体系结构、可视化软件建模、验证软件质量及控制软件变更等方面，针对所有关键的开发活动为每个开发成员提供了必要的准则、模板和工具指导，并确保全体成员共享相同的知识基础。它建立了简洁和清晰的过程结构，为开发过程提供较大的通用性。但同时它也存在一些不足： RUP只是一个开发过程，并没有涵盖软件过程的全部内容，例如它缺少关于软件运行和支持等方面的内容；此外，它没有支持多项目的开发结构，这在一定程度上降低了在开发组织内大范围实现重用的可能性。可以说RUP是一个非常好的开端 ，但并不完美，在实际的应用中可以根据需要对其进行改进。

* **在项目实践中采用了什么手段实现过程度量**



我们的软件开发实行的是PCDA方法，PDCA是英语单词Plan(计划)、Do(执行)、Check(检查)和Action(行动)的第一个字母，PDCA循环就是按照这样的顺序进行质量管理，并且循环不止地进行下去的科学程序。

1、P (plan) 计划，包括方针和目标的确定，以及活动规划的制定。计划职能包括三小部分：目标（goal）、实施计划（plan）、收支预算（budget）。

* 目标：在最初的阶段我们进行了需求分析，我们首先与用户进行了交涉，根据用户的描述，进行了初步目标制定，然后与客户商讨之后得出最终的的开发目标，即开发一款在线的,智能的,基于web的换脸工具，主要是不改变原头像的轮廓表情，换掉五官。
* 实施计划：然后我们将我们的项目分成了3次迭代，每次迭代制定了不同的计划，给每个人分配了任务。
* 收支预算:



2、D (Do) 执行,我们按照我们制定的计划分别在三次迭代中执行。

3、C (4C)，总结执行计划的结果，分清哪些对了，哪些错了，明确效果，找出问题。

Check（检查）：我们在每个故事点进行检查，主要检查团队成员的进度与任务的正确性，功能实现的完整性。

Communicate（沟通）：在每个迭代完成之后我们会与用户进行沟通，沟通的主要内容是上一个迭代任务的合理性以及下一个迭代的任务。

Clean（清理）：对于不合适的项目，我们会予以及时的清理以及改进。

Control（控制）：我们控制任务的正确性以及工作进度使之在规定的时间范围内完成。

4、A (2A)对总结检查的结果进行处理，对成功的经验加以肯定，并予以标准化；对于失败的教训也要总结，引起重视。对于没有解决的问题，应提交给下一个PDCA循环中去解决。通过度量发现了哪些问题，进行了哪些改进。

Act：我们对总结检查的结果进行了讨论以及改进。

Aim：对于目标不明确的任务或者目标出现偏差的任务进行了任务目标的重新制定。

* **通过度量发现了哪些问题，进行了哪些改进**

|  |  |
| --- | --- |
| 发现的问题 | 改进的办法 |
| 五官拼接功能与趣味贴图功能重叠 | 将五官拼接功能与趣味贴图合并为趣味贴图功能。 |
| tracking.js的人脸识别准确率太低，无法达到预期效果 | 将人脸识别插件由tracking.js更改为face++.js |
| 发现单独修改鼻子和嘴巴会显得人脸很不协调，往往是直接修改眼睛以下的面部更显得自然 | 将五官拼接中的鼻部切换和嘴部切换合并为面部切换 |
| 面相分析不太迎合年轻人的兴趣 | 将面相分析改为颜值测试 |