**开发团队年度绩效评估表**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名 | | 喻宙 | | 岗位 | 研发 | | | 评估时间 | | 2016/12/26 |
| 一、计划关键任务记录 | | | | | | | | | | |
| 序号 | 职责分类 | | 关键任务描述 | | | 其他参与人员 | 计划完成时间 | | 计划滞后比例 | 完成结果（是否有质量问题） |
| 1 | 开发  Portal 设置页面和访客端功能 | | 1. Portal 新版 Invitation 设置页面开发，合并了 Auto Invitation 和 Manual Invitation 的设置功能  2. 访客端新版本邀请功能的开发，邀请自动跳出聊天窗口 | | | Roy, Surey, Jason | 2016-02-22 | | - | 完成（无） |
| 2 | 开发  Customer Segmentation 后台设置页面 | | Segmentation 后台的设置页面开发，初步统一了 Condition 组件的前端部分。 | | | Surey, Roy | 2016-03-31 | | - | 完成（无） |
| 3 | 开发  访客端功能 | | 增加了访客端 ChatGroup 功能 | | | Surey | 2016-04-22 | | - | 完成（无） |
| 4 | 开发  访客端功能及后台页面 | | 1. 访客端发送消息限制功能 2. 访客端聊天后链接失效功能 | | | Allon | 2016-07-07 | | - | 完成（无） |
| 5 | 开发  History 后台页面 | | 后台聊天记录页面改进：   1. 根据新的聊天结构显示数据 2. 根据消息发送者不同显示不同颜色 3. 改进多行消息的显示 4. 增加聊天记录查询条件 rating comment | | | Surey | 2016-06-14 | | - | 完成（无） |
| 6 | 开发  手机端嵌入式聊天窗口 | | 增加手机嵌入式聊天窗口功能 | | | Surey | 2016-07-25 | | - | 完成（有）  受限于历史框架和开发周期影响，嵌入式窗口在部分老旧机型上没有很好的体验。 |
| 7 | 开发  New Branding 后台开发 | | 后台可以根据配置显示不同的语言文字及样式，一套程序可以给不同品牌运行 | | | Allon, Surey, Jason | 2016-09-17 | | - | 完成（无） |
| 8 | Condition 控件和 Dynamic Campaign 设置页面 | | ITSM 项目  统一后台 Condition 前端控件，适用于 Invitation、Routing Rules、Dynamic Campaign、Customer Segmentation 页面  开发后台 Dynamic Campaign 设置页面 | | | Lynn | 2016-09-23 | | - | 完成（无） |
| 9 | 开发  History 后台页面 | | 增加 Last Message Sent By 查询条件，修改后台以适应新的数据结构；  增加更多的 Priority 选择 | | | Surey | 2016-10-12 | | - | 完成（无） |
| 10 | 手机访客端发送文件 | | 增加手机访客端发送文件功能 | | |  |  | | - | 完成（无） |
| 11 | Porsche 访客端自定义开发 | | 使用 CustomJS + CustomCSS 为保时捷提供自定义化的访客端 | | | Mark, Roy |  | | - | 完成（无） |
| 12 | Web Agent Console | | 使用 React.js + Redux.js 开发基于 HTML 5 的新版 agent console 客户端 | | | Allon, Lynn, Mark, Andy, Allen, Davy |  | | - | 开发中 |
| 13 | 代码合并 | | 各类代码分支合并任务 | | |  |  | | - | 完成（无） |
| 二、2016年度工作小结（包括1、本年度主要任务与成绩；2、主要存在的问题、缺点及改进计划）  2016 年主要参与了前端相关的开发工作，包括访客端的开发与维护、后台页面的开发与维护，以及基于 HTML 5 的新版访客监控应用的前期调研与开发工作。  在完成分配任务之余，也致力于将自动化流程及现代前端工程技术引进到开发工作中。  针对当前开发环境及后台开发流程，编写了自动化脚本完成开发部分工作，如数据库切换、端口号切换、前端引用资源版本号替换等。简化了开发前期项目初始化及后期测试发布中的一些繁琐流程。  在访客端开发中，引入现代前端工程技术，重写了 newlivechat.js 代码，提供更清晰的模块和代码，更小的编译结果以及完整的单元测试。虽然代码本身没有在实际生产中被使用，但是为之后其他项目的构架提供了经验和实践支持。  在后台开发中，引入 React.js 技术栈。在不影响原有代码的基础上，渐进式的引入组件化的开发模式。目前已经统一了后台 condition 相关的控件代码，并提供相应的单元测试。  基于前期的开发实践经验，将现代前段工程技术引入到 HTML 5 版本访客监控的开发中，提供从前期开发到后期测试的完整工具链，包括代码风格检查，开发热替换调试，基于浏览器支持平台按需编译，单元测试，E2E测试等，优化了开发体验。  同时，调研了目前前端常用的 UI 框架，根据业务场景及后续业务发展方向，确定了以 React.js + Redux.js 为核心的开发构架。完成了新版访客监控开发的前期培训及部分代码开发。  主要存在以下几个问题：  1. CSS 相对薄弱。CSS 相关的基础不够扎实，CSS 部分相关的模块划分不如 JS 清晰。在维护老代码的过程中，CSS 部分产生的 bug 容易在修复的过程中产生反复。  针对这个问题，除了要补足基础知识的短板之外，也已经在项目中逐步推进 CSS Module的使用，用于解决 CSS 的先天缺陷，提供更清晰的组件化开发体验。  2. 重写的意愿明显强于重构。面对历史遗留代码，更愿意重写而不是重构。然而，相比于重构，重写会加长项目的研发周期，不利于后期测试。今年重写的 newlivechat.js 项目也并未真正投入生产。  针对这个问题，已经逐步在后台项目中尝试渐进式的前端开发模式，通过部分重构的方式逐步改进或替换原有代码。这一点，在 ITSM 的项目中已经有了初步的实践，会在之后的工作中继续尝试。  3. 修复 Bug 容易出现反复。在修复老代码的过程中，会出现 bug 反复的情况，如修复一个问题导致出现另一个问题，或者一个修复的问题在测试后期重新出现等。造成问题的主要原因，一方面在于修复问题时的考虑不够全面，比如只修复了问题描述中提到的情况，而没有考虑到其他相类似的情况；另一方面在于没有做足够的测试，对问题可能波及到的范围估计不足，出现修复一个问题引出另一个问题的情况。  针对这个问题，在所有新的项目中，都已经加入了单元测试的部分。虽然目前没有代码覆盖率方面的硬性指标，但是至少做到了测试提的 bug 修复后都附带有单元测试代码，保证当前修复的问题不会影响之前修复的问题，之后修复的问题也不会导致当前问题重新出现。  同时，在新版访客监控开发中引入了 E2E 的测试框架，计划通过 E2E 自动测试的方案，进一步完善代码的测试体系，减少可能存在的问题。 | | | | | | | | | | |
| 三、2017年工作目标与计划  1. 进一步完善和巩固基础知识。  2. 推进项目的持续集成。  目前公司的产品发布周期一般按月计，产品的更新换代非常缓慢，无法及时响应用户的反馈。计划进一步完善前端单元测试和集成测试的代码，同时打通从代码编写到发布的全套流程，争取做到至少每周发布新版本。用快速迭代的方式，来适应现在互联网社会快速发展的情况。  3. 前后端进一步分离。  实现后台 API 的 Mock，从而将前后端代码分离。分离解耦后的代码更轻量级，依赖更少，可以适应快速迭代的需求。同时，有利于开发的并行进行，加快开发的速度。  4. 前端优化  在将产品的中心放到基于 HTML 5 的 web 应用之后，前端有很多值得优化的地方用以改进用户体验，提升产品竞争力。 | | | | | | | | | | |
| 四、对工作、团队、公司发展、现有制度与管理等方面的意见或建议；  1. 开发的项目管理依然存在很多问题，一直以来都没有得到足够的重视和很好的解决。比如，代码合并周期太长，开发环境数据库资源稀缺等。工作流程亟待自动化脚本的大规模引入。  2. 团队整体的学术氛围不浓。对于新技术、新构架、新思潮的认识不足，思考不足，从长远来看不利于发展。  3. 产品构架有待改善，值得实践的方向包括：微服务化，前后端分离，Docker 等。 | | | | | | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 五、绩效评价（参照指标描述，在分数栏中打分，并按其分值相加计算出价评结果。） | | | | | |
| 评价项目 | 评价指标 | 指标描述 | | 自我评价 | 主管评价 |
| 工  作  完  成  质  量  40分 | 代码格式  （10分） | 代码非常清晰，简洁易懂。（8－10） | | 9 |  |
| 代码比较清晰，可读性尚可。（4－7） | |
| 代码条理不清，可读性较差。（1－3） | |
| 代码  逻辑性  （10分） | 代码具有逻辑性，流程最简化，依赖性小。（8－10） | | 9 |  |
| 代码逻辑性尚可，但有一定依赖性。（4－7） | |
| 代码逻辑不强，可能存在维护性风险。（1－3） | |
| 代码  可用性  （10分） | 代码实现合理，效率较高，可保证软件运行。（8－10） | | 8 |  |
| 代码实现基本合理，软件运行速度一般，。（4－7） | |
| 代码实现不太合理，效率较低，影响软件运行。（1－3） | |
| 代码  复用性  （10分） | 代码可多次运用，复用性优秀（8－10） | | 9 |  |
| 代码只能少量运用，复用性一般。（4－7） | |
| 代码完全不能复用。（1－3） | |
| 工  作  难  度  20分 | 项目  重要性  （10分） | 大型项目，项目优先级高。（8－10） | | 7 |  |
| 一般项目，项目重要程度一般。（4－7） | |
| 小型项目，如Bug修复之类。（1－3） | |
| 专业  复杂度  （5分） | 项目非常复杂，有很大的挑战。（5） | | 4 |  |
| 项目难度一般，有一定的挑战性。（3－4） | |
| 小型项目，无难度。（1－2） | |
| 工作  创新度  （5分） | 具有创新思维，不断思考优化开发工作。（5） | | 5 |  |
| 有一定的创新，可以提出一些创新性建议（3－4） | |
| 按部就班，较少思考创新。（1－2） | |
| 工作量  10分 | | 工作量很饱满，在完成开发任务的同时，还承担了项目之外的工作。（8－10） | | 9 |  |
| 工作量饱满，基本完成项目开发任务。（4－7） | |
| 工作量不饱满，尚有空闲。（1－3） | |
| 工  作  态  度  30分 | 主动高效  （10分） | 独立提出切实可行的改进方案，并推进实施，取得良好的成效。（9－10） | | 8 |  |
| 工作中主动发现问题，提出有价值的改进建议。（7－8） | |
| 调动各方面资源以达成目标。（4－6） | |
| 被动执行安排的工作，遇到困难被动等待，对工作中问题视而不见。（1－3） | |
| 团队协作  （10分） | 协助对方获取成功，在达成团队整体目标的同时实现个人目标。（9－10） | | 8 |  |
| 认真听取对方意见，清楚表达自己意见，并且提出有价值的建议。（7－8） | |
| 与团队成员良好沟通，保证项目按期进行。（4－6） | |
| 不与团队成员沟通，完全按照个人设想工作。（1－3） | |
| 学习分享  （10分） | 主动学习，工作技能明显提高，分享并帮助团队成员进步。（9－10） | | 10 |  |
| 有意识地学习知识技能和业界先进经验，并在工作中实践，乐于分享。（7－8） | |
| 能够参加公司培训，在某些方面提高自己，有一定的分享。（4－6） | |
| 缺乏自我培养和提高意识，随大流。（1－3） | |
| 自评结果： 86 | | | 主管评分： N/A | | |
| 主管反馈：包括继续发挥的优点和下一步努力方向。  （若评价指标自评与主管评分差距较大，建议在此列出具体的案例来说明问题，以便于后续沟通。）  主管签名： | | | | | |