**技术原型迭代计划**

项目名称：交集——校园活动集成平台

组号：第9组

小组成员：胡彤 杨菡雪 全雨乐 张奕涵 杜心敏

迭代名称：技术原型迭代

制定日期：2023年3月19日

计划起止日期：2023年3月20日至2023年5月16日

**1. 进度安排及人员分配**

**2. 任务**

**技术方案设计**

* 架构风格选择  
  前后端分离，需要确定前端框架、后端框架和数据库框架
* 架构视图设计
* 主要功能关键算法设计  
  推荐算法的选择和设计
* 编程规范学习确定  
  对编程过程中所用到的每种语言，均采用google开源项目代码规范，具体规范内容待学习

**技术原型实现**

**前端**

* 各个界面的JS实现  
  主要为：登录界面（接入jaccount）、活动列表、活动详情、管理活动、交友圈、站内私信、用户主页
* 界面风格统一  
  对于前一阶段迭代的界面风格不统一等问题进行弥补，保证新的界面风格统一、逻辑清晰
* 跳转逻辑设计  
  保证跳转步数少，简单易用

**后端**

* 用户数据库和userService搭建  
  搭建可靠的数据库存储用户名和密码，以及可能存在的其他用户状态
* 活动数据库和ActivityService搭建  
  根据已有活动搭建活动数据库
* 前后端交互  
  保证前后端通信的安全和流畅

**测试与评审**

* 撰写和评审软件架构文档
* 测试技术原型并改进
* 编写迭代评估报告

**预期成果**

* 初步可用的登录页面
* 活动列表、活动详情、管理活动
* 交友圈、站内私信、用户主页
* 迭代评估报告

**3. 主要风险**

**风险1：技术风险**

软件开发项目是一种技术密集型项目，重大的技术风险包括：

1. 软件结构体系存在问题，使完成的软件产品未能实现项目预定目标;
2. 项目实施过程中采用全新技术，由于技术本身存在缺陷造成开发出的产品性能以及质量低劣的问题
3. 软件开发过程中需要运用项目组缺乏经验的技术或专有技术

这些技术问题问题决定了项目的进度、质量，甚至成败，因此对软件项目中识别到的技术风险要有足够的重视，并采取积极的措施加以应对。

应对方案：

1. 实事求是：项目组一定要本着项目的实际要求，选用合适、成熟的技术，千万不要无视项目的实际情况而选用一些虽然先进但并非项目所必须且自己又不熟悉的技术。如果项目所要求的技术项目成员不具备或掌握不够，则需要重点关注该风险因素。在技术应用之前，需要针对相关人员开展好技术培训工作，确保团队的成员熟练掌握这些技术。
2. 慎重选择开发中运用到的技术：例如在我们的软件工程原理与实践这门课的大作业里，我们需要慎重选择编程语言，界面设计也需要我们选择合适的平台，选定的技术要成熟，不能有很大的漏洞，也不应太过于小众或晦涩，以免出现技术问题时不方便解决。
3. 平衡技术创新与技术风险：对新技术的使用要谨慎，团队应尽量采取成熟的技术方案完成软件开发工作。不能一味地追求技术上的创新，还应该考虑到技术上的风险。若团队希望在技术创新方面做的出色，就应该做好技术风险高的准备，因此要慎重对待创新与风险的平衡，以免造成项目推进到半程做不下去的情况，即费时又消耗人力，得不偿失。

**风险2：进度风险**

软件开发常常对项目工期有着严苛的要求，软件进度的延迟往往意味着违约或市场机会的缺失。课程中的软件开发也有相应的时间限制，项目进度控制出现问题则会导致课程任务无法及时完成，面临挂科风险。软件开发项目中影响进度的因素很多，如人为因素、技术因素、资金因素、环境因素等等。在软件开项目的实施中，人的因素是最重要的因素，技术的因素归根到底也是人的因素。

应对方案：

1. 制定项目计划。开始时的项目计划可以先制定得比较粗一些，随着项目的进展，特别是需求明确以后，项目的计划就可以进一步的明确，这时候应该对项目计划进行调整修订，通过变更手续取得开发人员的共识。
2. 项目开发过程中进行必要的测试。必要的测试是项目渐近明细的方式之一，随着项目的推进再进一步细化、调整、修正和完善。
3. 开发过程及时更新相关文档，记录项目进度、当前任务、版本管理等。软件开发组织在工期的压力下，往往放弃文档的编写与更新，结果在软件项目的晚期大量需要通过文档进行协调时，却拖累软件进度越来越慢。
4. 持续地监控，及时纠正实际进度与计划的偏离。项目进度控制是随着项目的进行而不断进行的，是一个动态过程，也是一个循环进行的过程。在计划制定时就要确定项目总进度目标与分进度目标；在项目进展的全过程中，进行计划进度与实际进度的比较，及时发现偏离，及时采取措施纠正或者预防，协调项目参与人员之间的进度关系。

**风险3： 安全风险**

软件属于逻辑产品，很多情况下并非由于软件失效导致不安全情况出现，而是在软件正常工作时，在某种特殊条件下软硬件相互作用或由于人的使用问题导致异常情况发生。凡是与软件相关的接口、硬件状态、系统时序、人员操作、使用环境、软件自身的逻辑均属于应考虑的软件安全风险的范畴。

风险：

1. 信息泄露：随着网络化建设，用户和设备及应用程序和数据正在向私人控制区域之外迁移。这意味着，如果有人拥有正确的用户凭据，则他们将被允许进入他们请求的任何站点、应用程序或设备。这导致暴露的风险增加，从而瓦解了曾经值得信任的用户控制区域，并使许多用户面临数据泄露、恶意软件和勒索软件攻击的风险。
2. 恶意攻击：系统可用性既包括可靠性、软件复制、灾难恢复等，也包括可用性相关的安全性内容，保护系统使其免受降低可用性的恶意攻击。这样的恶意攻击叫做拒绝服务（Denial Of Service）。此类攻击会让软件某一环节无法正常运行，产生严重的后果。

应对方案：

1. 身份准入：基于严格身份验证过程，即只有经过身份验证和授权的用户和设备才能访问应用程序和数据。在本次项目中，以jaccount接口限制用户登入身份，保证用户可靠性，以此减少身份信息及其他个人信息泄露的风险。
2. 确保信息保密性：不在永久性 cookie 中存储敏感数据，不使用 HTTP-GET 协议传递敏感数据，记录详细的错误信息，及时捕捉异常现象。
3. 保障最弱环节的安全性：识别并改善系统安全性的最弱环节，直到安全性风险达到一种可接受的等级。
4. SQL注入防范：进行数据库操作的时候，对用户提交的数据必须过滤 ’ ; -- 等特殊字符。
5. 保护可用性：应能设置系统会话时间，防止会话劫持和重复攻击的风险。对于高度保护的应用系统，可将超时时间设置为5分钟，低风险的应用系统设置不能超过20分钟。