前后端分离：

1. 整体项目我们采用的是前后端分离的开发方式，通过这种方式我们可以在开发的过程中专注于前端技术或后端技术单方面的学习和开发，提高效率。
2. 从项目的角度来看，通过前后端分离的开发方式可以实现真正的前后端解耦，减少将来业务流量增加带来的服务器负载。
3. 在测试的时候，也可以更快的找到BUG的出处，分别交由前端或后端的工程师来进行修改。
4. 如果由于过大的流量导致服务器宕机，也不会对前端页面造成较大影响。同时由于采用的异步加载，不管前端的页面多复杂也不会影响服务器的响应速度。

前端技术：

1.在前端基础框架方面，我们选择使用了VUE框架，这种框架的优势在于它的核心库只关注视图层，在和后端的链接方面有较大的优势。

2.响应式编程是VUE的一大特点，利用Vuex建立的Store（仓库），当组件从store中提取状态信息的时候，如果状态信息产生了变话，相应的组件也会相应的得到高效更新。这使在调试的过程中可以实时的对修改结果进行监控，提高开发效率。

3.组件化是VUE的另一大特点，利用iview的组件库，我们可以依据客户的需求进行实时的修改。针对保险理赔业务需要大量文字描述的需求。

4.在我们的框架中还集成了vue-quill-editor通过它来实现文本的输入

前后端连接：

我们使用一个基于promise的HTTP库——Axios来实现前后端的数据交互。他可以通过Get和Post方法向后台发送数据。在请求过程中还可以实现很多的可控功能，比如拦截器可以在请求或者响应时拦截HTTP请求来进行处理。同时还能防止XSRF（Cross-site request forgery）攻击。

后端技术栈：

1. 我们使用spring boot框架进行后端的开发，它可以简化新Spring应用的初始搭建以及开发过程，通过导入“starter”模块，可以将许多的依赖添加到工程之中。

2. 我们使用持久层框架mybatis，通过将sql语句写在xml文件中，来解除sql与程序代码的耦合，便于统一管理和优化，提高了可维护性。

3. 为了保证应用的安全性我们采用了Spring security框架，其中包括了用户认证（Authentication）和用户授权（Authorization）两个部分。其框架本身以及spring框架带来的灵活性，能够很好的满足我们应用开发的典型需求。

4. 同时，我们还将配合Spring Security安全框架使用OAuth2，来实现未授权客户访问受保护页面的时候的网页重定向，同时通过它我们还实现了github和qq账户授权登陆服务。

5. 我们还将使用redis来缓存数据，以缓解大规模用户同时访问时对服务器和mysql的压力。

milestones：

在项目开发的过程中，我们项目组选用的是螺旋开发模型，在多次迭代的过程中插入风险分析和客户评估。减少开发过程中风险造成的损失。通过每次迭代加入的客户评估及时对开发方向进行调整。

前端部分：

1. Week6：在本周依据客户的需求完成了前端页面基础框架的构建，在风险评估的阶段，预估了将来和后端对接时会产生的问题，同时准备了一套备用框架作为备选方案。
2. Week7：第一次迭代，设计了满足业务需求的链接跳转，经过风险评估，确定现有方案可行，并进行了开发，同时计划第八周的前后端对接测试
3. Week8：第二次迭代，第八周前后端对接测试发现现有方案在与后端对接的时候无法链接，决定采用备用框架，实现前后端信息交互。同时决定延后一周进行首页和其他页面的背景和细节美化，计划生成多方案交由客户评估。
4. Week10: 计划进行第三次迭代，加入背景，同时对边框按钮输入框等位置进行细节美化。评估这部分美化可能带来的风险为客户对配色风格的不满意，准备多种风格的页面交由客户评估，计划下版本的迭代加入安卓前端的开发
5. Week11: 计划进行第四次迭代，对安卓端app的基础页面开发过程进行设计，对安卓端的风险进行评估。
6. Week12: 进行安卓端和后端的链接，评估链接过程中可能的风险，完成后交由客户评估。
7. Week13: 对整体项目进行最终测试。