**技术选型与架构**

**DataBase**

采用mysql，建三张表：用户表、好友关系表、信息发送表

**用户表：**

用户id号：int类型（主键）

用户账号：字符串类型

用户密码：字符串类型

用户昵称：字符串类型

用户性别：字符串类型

**好友信息表：**

用户id号：int类型（主键）

好友id号：int类型（外键）

**信息发送表：**

信息发送方id号：int类型（主键）

信息接收方id号：int类型（外键）

发送时间：datetime类型

发送内容：blob类型

**Client**

客户端的开发采用MVC设计模式，基础框架就是Foundation和UIKit，以后看情况再选择使用别的框架

**User**

用一个User类来封装客户端用户的数据：

1、用户账号 NSString

2、用户昵称 NSString

3、用户性别 NSString或枚举

4、好友列表 NSMutableArray

5、聊天记录路径 NSURL或者NSString

**ViewController**

以UITabBarController为根控制器，再设置三个标签页，然后以这个UITabBarController为容器，添加三个导航控制器UINavigationController，然后再以每一个导航控制器为容器，push和pop所需要的ViewController。

**View**

这里的话，我们小组因为学习得还不够透彻，所以打算进一步学习后，再根据交互设计文档进行设计，同时根据学习的内容，反馈来优化改良交互设计文档。不过这里我们小组暂时是讨论到了要使用UITableView、UIButton、UITextView等控件和视图，但因为对布局这些暂时学得还不是很深，故没有具体设计，最后还会考虑是否要添加一些手势控制之类的功能

**Model**

**FileReadOrWriteModel**

**NetworkModel（应该会进行分类）**

**1、注册功能：**

参数：NSString 账号，NSString 密码，NSString 昵称，NSString 性别（昵称和性别看情况要不要使用枚举）

返回：NSString @"注册成功" 或者 @"注册失败"

**2、登录功能：**

参数：NSString 账号，NSString 密码

返回：NSString @"登录成功" 或者 @"登录失败"

**3、下线功能：**

参数：NSString 账号

返回：NSString @"下线成功" 或者 @"下线失败"

**4、发送信息功能：**

参数：NSString 发送方账号，NSString 接收方账号，NSData 二进制数据流（文字或图片）

返回：NSString @"发送成功" 或者 @"发送失败"

**5、接收信息功能：**

参数：NSString 当前用户账号

返回：NSArray 当前用户在数据库中所要接收的信息封装成一个数组返回

说明：这个NSArray数组中的元素类型为NSDictionary,存储发送方账号(key)和信息内容(value)

**6、添加好友功能：**

参数：NSString 当前用户的账号，NSString 所要添加好友的账号

返回：NSString @"添加成功" 或者 @"不存在该用户"

**7、删除好友功能：**

参数：NSString 当前用户的账号，NSString 所要添加好友的账号

返回：NSString @"删除成功" 或者 @"你没有这个好友"

**8、获取好友列表功能：**

参数：NSString 当前用户的账号

返回：NSMutableArray 当前用户在数据库中的好友信息封装成一个可变数组返回

说明：这个NSMutableArray数组中的元素类型为NSArray,存储好友的账号、昵称、性别

**9、获取个人信息功能：**

参数：NSString 当前用户的账号

返回：User 这是客户端中设置的一个类，存储用户的账号、昵称、性别

**10、修改个人信息功能：**

参数：NSString 账号，NSString 昵称，NSString 性别

返回：NSString @"修改成功" 或者 @"修改失败"

**Server**

用Java EE、Tomcat来部署服务器

采用简易的MVC设计模式(Model 2)，以Servlet为Controller，Service为Model，不过没有View了

以redis作缓存

以json的信息格式与客户端交互数据

JDBC来连接数据库