北京理工大学计算机学院
《Java 程序设计》课程设计
MyForum 论坛应用设计

班级 07112303 学号 1120232667 姓名 伍奕涛

一、应用开发介绍

1. 开发原因

考虑选题的时候高中同学群里用腾讯文档搭建了一个简易的班级留言板，但是使用起来比较麻烦，所以选择开发一个能够支持两位数级别用户数量的论坛应用（实际上也考虑了一些高并发的极端情况，但肯定比不上成熟的网络应用）。

2. 开发结果

MyForum 是一个半成品，并没有实现设计的全部功能，甚至一些基础的功能都没有完整实现（部分功能的前端部分没有完成，比如评论功能）。造成这个结果的主要原因如下：

（1）整个项目在前期缺少详细严谨的设计，这导致了我对这个项目的复杂程度产生了误判（论坛的复杂程度超出想象），而且经常返工；（2）我希望通过这个项目加强我的面向对象设计能力，因而大多数功能我都是从底层开始设计实现，除了一些超出能力范围的功能外，其他的完全自己实现。这增大了工作量，带来了许多错误，但是我也积累了一些经验。

作为一个论坛软件，连基本的评论功能都没有实现的确是致命的不足。但是最后我还是选择将剩余的时间放在设计文档上，这样做的理由是评论功能的后端部分已经完整实现，且应用已经实现了发布、获取和展示帖子等一系列功能，而评论功能未实现部分的逻辑与此完全一致。同时我尽可能地提高了代码的可拓展性，以方便后续对功能的完善与拓展，也尽可能的考虑并解决高并发带来的问题。

3. 已实现的功能和计划功能

（1）服务端与客户端、服务端与数据库的通信
（2）登录与注册
（3）异地登录占线
（4）发布帖子
（5）获取帖子
（6）帖子修改（前端未实现）
（7）删除帖子（前端未实现）
（8）评论与点赞（前端未实现）
（9）用户信息管理（前端未实现）
（10）管理员系统（未实现）

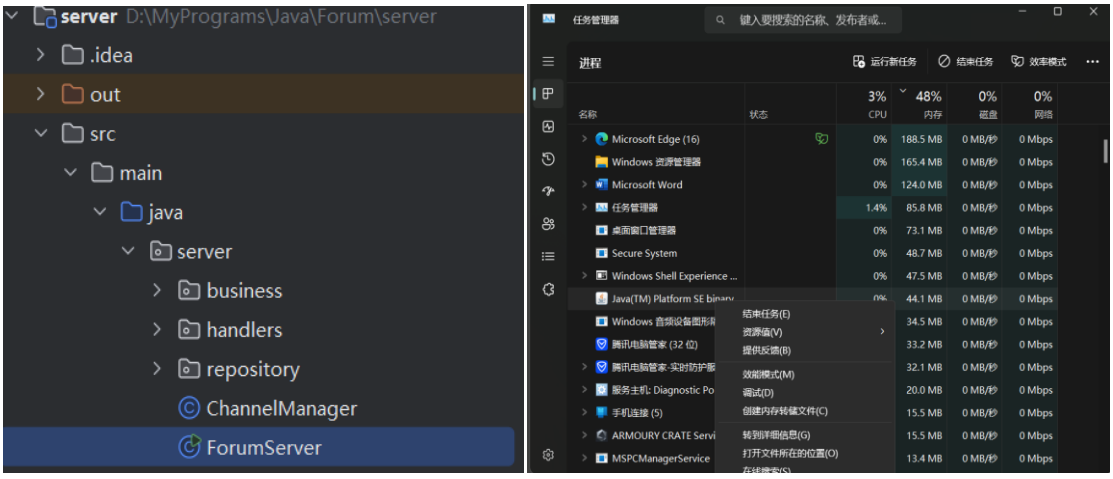
二、应用的运行环境、运行步骤

1. **运行环境：**JDK 21
2. **程序组成：**基础模块 base、客户端模块 client、服务端 server，邮件模块 mail。其中 client.jar 是客户端模块可直接执行的 jar 包，server.jar 是服务端可直接执行的 jar 包（极其不建议直接运行 server.jar，因为没有为它构建可视化界面以及便捷的关闭方法，直接运行会导致程序在后台隐式运行,而且数据库也在，建议在 IntelliJ 终端中运行），剩余模块的 jar 包均在相应目录的 .\out\artifacts 路径下。
3. **运行步骤**

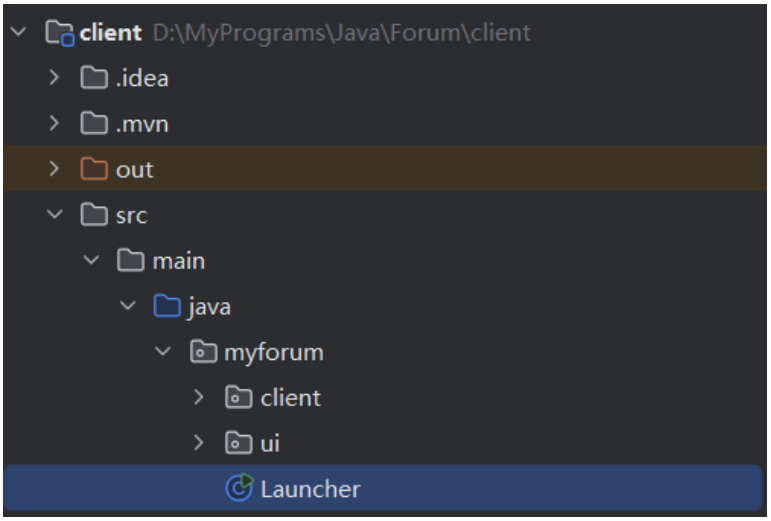
（1）安装 JRE 21。

（2）服务端程序默认监听本机 8080 端口，请保证 8080 端口可用。若 8080 端口不可用，请使用 IntelliJ 打开 server 文件夹，打开 server 包下的 ForumServer，将其中的 port 字段设置为可用端口。

（3）运行服务端程序，使用 IntelliJ 打开 server 文件夹，通过运行 server 包下的 ForumServer 来运行服务端程序。如果直接双击运行了 server.jar，可按下 Windows 快捷键 Ctrl+Shift+Esc 打开任务资源管理器，通过关闭 Java 虚拟机来结束程序。（如果模块依赖有问题，之后的内容中有模块间的依赖关系）



（3）如果没有修改服务端的监听端口，或者服务端程序与客户端程序在同一台计算机上运行，请跳过此步。如果修改了服务端程序的监听端口，请使用 IntelliJ 打开 client 文件夹，打开 myforum 包下的 Launcher，将 main 方法中调用的 ClientApplication.main 方法中的第二个参数修改为服务端中设定的端口号。如果服务端程序与客户端程序不在同一台计算机运行，请将第一个参数修改为运行服务端程序的计算机的当前 ip 地址。



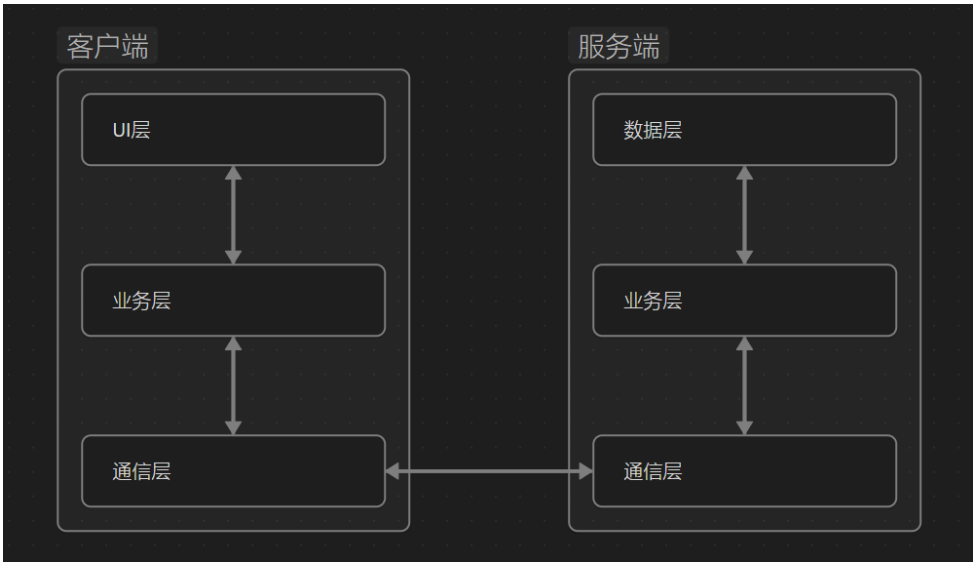
（4）运行客户端程序，双击 client.jar 运行客户端程序。
（5）如果不想注册账号可以使用账号 123@.1 ，密码 123。

三、程序开发平台

1. 代码行数：约 4000 行（已实现功能的行数）。
2. 开发环境：IntelliJ 2024.2.1 + JDK 21、SQLite 3.39、JavaFX Scene Builder 23.0.1

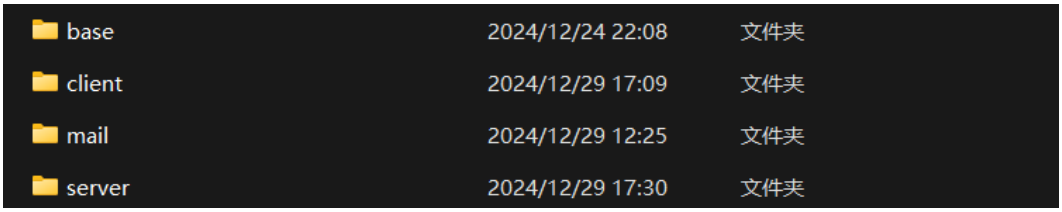
四、应用架构

1. 逻辑架构

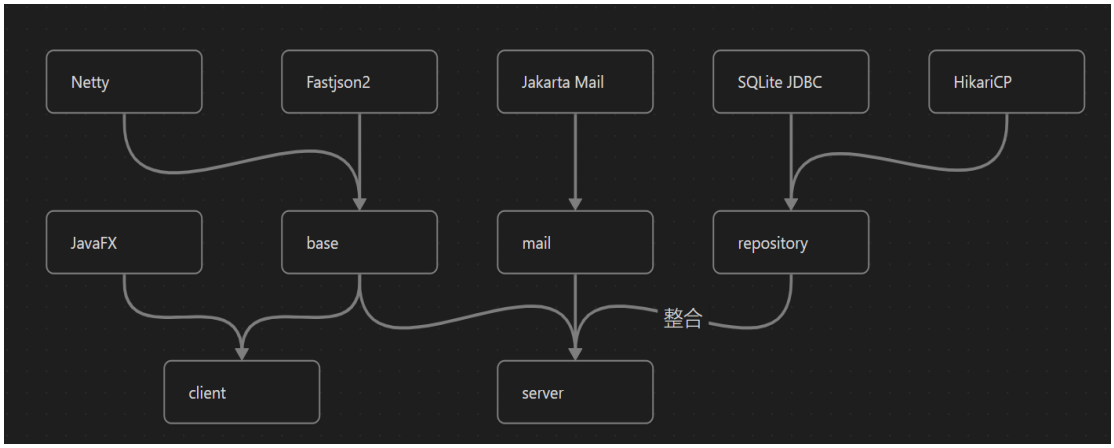


2. 代码架构

整个应用由以下四个部分的代码组成。



代码的依赖关系如下。



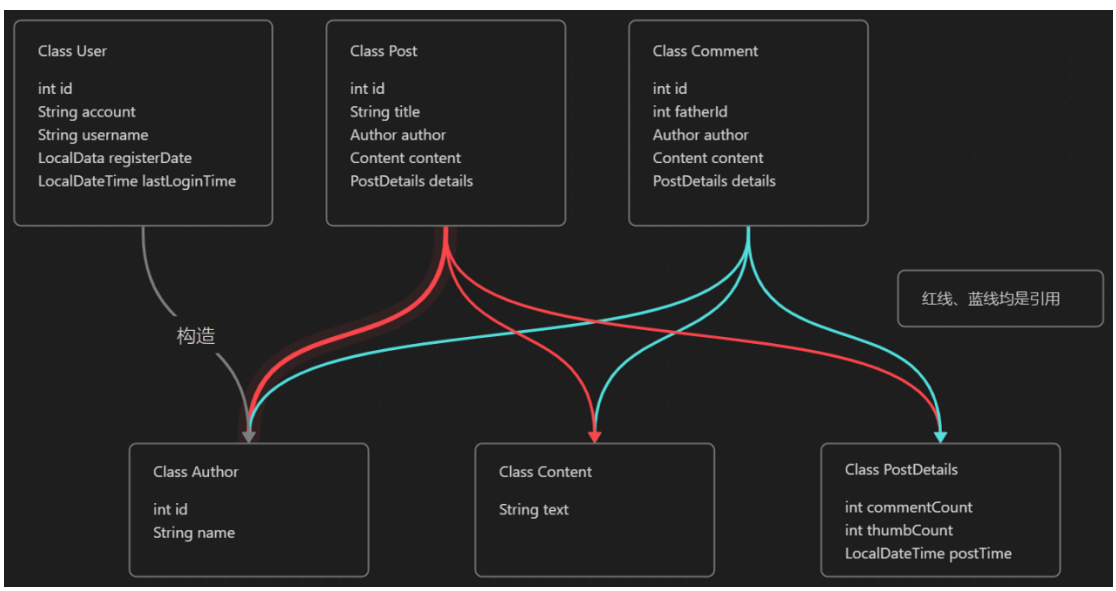
五、各模块详情

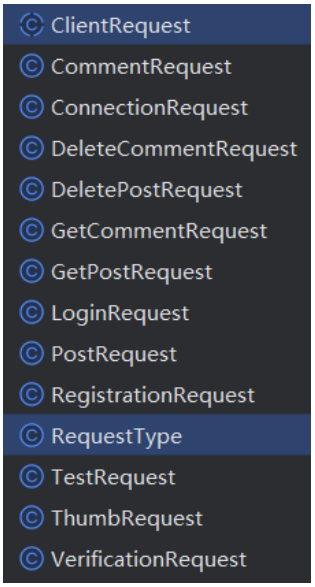
1. base 模块

base 包是整个应用的基础包，里面构建了应用的基础数据结构 model 包、通讯协议 request 包与 response 包和解码译码工具 Packet 包。

model 包中抽象了应用业务层中的三个主要数据结构 User、Post 和 Comment，并在此基础上构造了其余的辅助类。其中 Content 类设计的主要目的是方便拓展帖的形式，帖和评论目前只支持短文本，在未来加入图片、视频等后，Content 类的存在能减少业务层代码的变动。Author 类避免在 User 类的属性增加后，Post 和 Comment 存储额外的不必要信息。PostDetails 类可以在未来存放 Post、Comment 需要增加的属性，避免与 Post 和 Comment 直接相关的方法做较大改动。

通过 id 和 fatherId 属性，一个 Post 和与之相关的 Comment 组成了一棵多叉树。





request 包和 response 包在功能上完全对偶，唯一不同是 response 包中的对象多了一个表明请求是否成功的 boolean 属性 success，因此只介绍 request 包。

RequestType 类中存贮了客户端到服务端的操作码，实现了通过操作码返回相应操作类型的方法，这是课程 ppt 中的内容。但是在实践中感觉这个类不是特别好用，因而使用了其他方法来确定操作对象的具体类型，之后会计划删掉这个类。

ClientRequest 是一个抽象类，也是包中剩下的类的基类，其中定义了两个抽象方法 getRequestCode 和 getRequestType，前者返回操作码，后者直接返回 ClientReqeust 对象在被创建时的类型，因而可以代替 RequestType 类的功能。其余的类都是对客户端向服务端发出的请求的抽象。

Packet 包使用 Fastjsaon2 技术，是课程 ppt 上的内容精简后的结果，因此不再赘述。

2. server 模块

server 模块在开发初期是两个模块 server 和 repository。server 模块实现服务端通信层和业务层，repository 实现数据层。但是后期发现添加依赖会导致数据库不可用（可能是因为将数据库也打包进 jar 包了，但需要在包外操作数据库），因而最后将两个模块合并。

repository 包是对 JDBC 的封装。数据库设计中，设计了 user、register、post 三个表，分别记录用户信息，注册信息（这两个表的 id 是对应的）和帖与评论（由于存在评论的评论这种情况，帖和评论的 id 表应该是共享的，放在一个表里方便了对 id 的管理），但是这三个表放在同一个数据库中，带来了一些性能缺陷。对于数据库的连接，考虑到高并发情况下数据库操作的线程安全，引入了外部库 HikariCP，这是一个用来管理数据库连接的库，通过连接池管理数据库连接，一个数据库连接在同一时

刻只能被分配一次，从而达到保障数据库数据的线程安全的目的。但因为 MyForum 只有一个数据库，因而连接池会造成阻塞导致性能降低，之后应该会将这三个数据表拆为三个数据库。

同时在数据层应用了 DAO 模型，将数据库的数据结构抽象为 model 包中的类，将每一个数据表中的操作抽象为一个相应的 manager 类，其中同样也有继承的面向对象思想。

handler 包与独立于其他包的 ForumServer 构成服务端的通讯层，这一层的配置也与课程 ppt 中相差无几。主要是管线的构建和连接，实现了不定长发报收报，心跳与保活（接收和发送 ConnectionRequest 和 ConnectionResponse），和业务处理。在业务处理中，为每一个 Channel 配置了一个用于执行业务的线程池，将通讯与业务处理的线程分离，避免阻塞。

```
pipeline.addLast(new LengthFieldBasedFrameDecoder( maxFrameLength: 1
pipeline.addLast(new IdleStateHandler( readerIdleTime: 10,  writerIdleTim
pipeline.addLast(new ClientRequestDecoder());
pipeline.addLast(new HeartbeatServerHandler());
//测试连接用Handler
  pipeline.addLast(new ConnectionTestHandler());
pipeline.addLast(new ForumServerHandler());
//出站管线
pipeline.addFirst(new ServerResponseEncoder());
pipeline.addFirst(new LengthFieldPrepender( lengthFieldLength: 4));
```

在业务 Handler 中，值得一提的是对业务分发的解决方案 ForumServerHandler，这里应该体现了我目前对于面向对象思想的全部理解以及掌握情况。

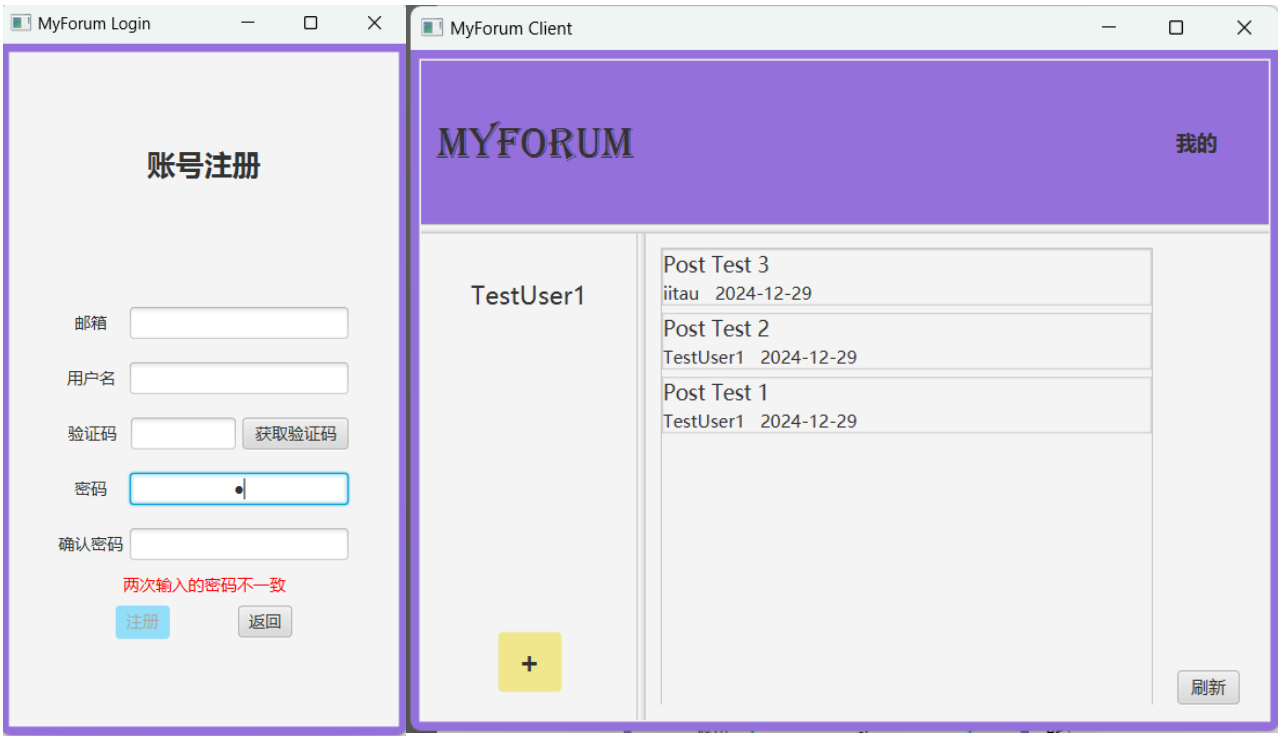
由于 Java 中方法的返回对象不能是动态的，所以所有从 Channel 中接收到的数据在解码后其当前数据类型都是 ClientRequest，因而首先需要将数据类型进行强制类型转换。在 MyForum 中，服务端调用 ClientRequest 对象的 getRequestType 方法来获取该对象在被创建时的数据类型，然后通过反射实现动态强制转换数据类型。然后调用业务分发器 Dispatcher 通过对象的 getRequestCode 方法定位到相应的业务处理方法数组里的一个方法（现在数组里存的是一个特定对象，这个对象只有一个 response 方法来执行业务操作，之后应该会完善为一个函数式接口数组），并将其作为参数传入。

因为业务处理方法的参数不定，对业务处理方法的构建使用了泛型。具体的业务操作过多，就只介绍占线功能了。在数据库 register 表中存储了当前账户的登录情况（之后可以增加“冻结”状态），在 ForumServer 中存储了在线的账户 id 与 channel 的映射，当一个账号登录时会检查数据库中的账户登录状态，如果此时账号已经登录，则会在服务端映射中查找与当前账号 id 绑定的 channel，并将之关闭。

3. client 模块

client 模块的通讯层和业务层与 server 模是对偶的，因而只介绍 JavaFX 部分，如果一种技术有多次应用也只介绍其中最经典的。

在注册界面中，通过监视实现只有密码和确认密码完全一致才会解锁注册按钮，否则显示相应提示；在主页面中，右下部分是动态生成的，方便动态添加 Post，每个显示 Post 的控件都添加了鼠标点击响应（是可以点进去看的!），上面的标签暂时不可点因为没做完。。。之后会添加用户信息界面，点击“我的”后动态更新右下部分而不是弹出另一个窗口。点击+号可以发帖（很丑，因为不知道怎么在做成圆形的同时能居中而且在窗口大小变动的时候仍然是一个标准的圆）。



六、亮点与不足

1. 亮点

（1）ForumServerHandler 中业务分发问题的解决方案

（2）数据层使用 DAO 模型

（3）JavaFX 中使用 MVC 架构

（4）应用是多线程的，而且线程基本安全（应该是的，至少目前没有出错。。。）

（5）异地登录占线功能

（6）应用运行时终端有清晰详细的信息输出

（7）客户端和服务端关闭后的处理周全，不会有资源泄露

（8）90%的代码都是自力更生的（我觉得在软件开发学习阶段这应该能算亮点吧）

2. 不足

（1）应用是半成品，评论功能没做完

（2）没有一开始就做好计划，返工浪费了很多时间

（3）注释少且不详细，也因此浪费了很多时间与精力，应该向 JDK 的源码

学习

（4）没有做全面的测试

（5）时间规划太不合理了，这篇文档开始写于 2024 年 12 月 29 日下午 5 点左右，其时刚好完成现有应用的初步调试，结束于 2024 年 12 月 29 日 23：55

七、课程建议

课程内容很多，但质量也很高而且贴近于实际的应用，因而显得学时有些紧，有些内容还没学透（比如函数式编程，在 MyForum 的开发中完全没有应用）。感觉这门课更应该叫《Java 开发技术》而不是《java 程序设计语言》，这的确在选课时误导我了，但我也因此出乎意料地学到了很多实用的技术。