数据库系统与应用 Lab3 导引

罗永平，ypluo@mail.ustc.edu.cn

Lab3 作为一个真正的数据库应用开发实验，具有需求/功能分析，架构和语言选择，UI 设计和后端逻辑实现，软件测试，撰写(说明)文档等一整套流程，能够充分锻炼和考察学生的软件工程能力。

为了让大家对软件开发有基本的了解，并且从无到有开发出一个银行数据管理系统，我们需要先介绍一些基本的概念。

前后端

- 在软件开发中，前后端分别指的是显示层（前端）和数据处理层（后端）。前端负责并接受用户的输入，并将处理结果展示给用户看（User Interface），后端负责解析用户需求并执行。

- 前端直接和用户打交道，后端和前端以及数据打交道。当前后端完全根据双方预定义的 API 进行数据和控制的交互时，也就实现了前后端功能的分离。
- 有时候后端可能需要和广义的数据库打交道，如果将数据存储模型以及数据存取过程从后端中分离出来，那么就有点类似 MVC（Model-View-Controller）框架了。这样一个软件开发便变成了图形界面 + 数据存储和存取 + 数据和控制的派遣三个部分了。
- 有些同学可能会纳闷，分这么多有什么用呢？ 其实在大规模的软件开发过程中，合理的进行责任划分能够将庞杂的问题拆分成多个部分，让不同的人去做自己擅长的部分。此外，合理的拆分能够使代码模块化更强，方便测试，复用和阅读。

架构的选择

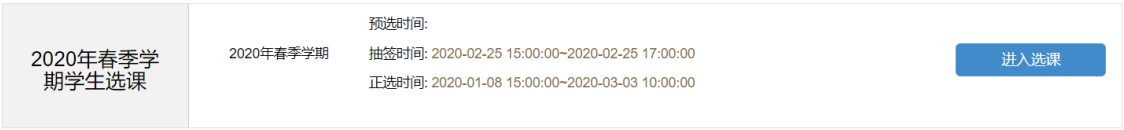
一个互联网应用，其后端就不得不与服务器挂钩。实现一个互联网应用通常采用如下两种架构：C/S 架构和 B/S 架构

C/S 架构

- C/S 架构即 Client/Server 架构，其中 Client 是一个运行在用户设备上的软件，它负责接受用户的请求，显示处理结果，必要的时候还能完成一些简单的计算。Server 是一个运行在服务器主机的一整套服务，它负责接受用户的请求，并从数据库中获得请求的数据结构并返回给客户端。
- Client 和 Server 通常在传输层工作，使用 TCP 和 UDP 协议进行通信。
- 采用 C/S 架构的常见例子：QQ，微信的手机端和电脑端

B/S 架构

- B/S 架构即 Browser/Server 架构，Browser（浏览器）取代了 Client，负责图像界面的功能，此处的 Server 它同样使运行在服务器主机的一整套服务，但是其与 Browser 交互的对象不再是数据包，而是 http 请求和 html 网页。
- Browser 和 Server 在应用层工作，使用 http 协议进行通信。
- 采用 B/S 架构的常见例子：QQ 网页版，科大选课系统



C/S 架构和 B/S 架构适用的场景不同，其优缺点也不同。但对于本次实验而言，选择两种架构都是可行的。值得一提的是：

选择 C/S 架构最直接的优点就是不需要浸染复杂的 web 框架和网页设计。

选择 B/S 架构最直接的优点是用户界面设计更漂亮，可以方便的开发子页面，且天然支持跨平台。

选择语言

无论采用何种软件架构，都有丰富的编程语言可供大家选择，因此可以根据自己熟悉的语言开发。

python

- C/S ： python + pyQt/Tkinter（python 下的图形界面开发库）
- B/S：python + Flask/Django（web 框架）

CPP

- C/S：cpp + QT（最热门的图形界面开发库）
- B/S：cpp+ CPPCMS/wt （其他 cpp 向的 web 框架）

Java

- C/S: java + Swing （java 图形界面库）
- B/S: java + jsp （java 服务器动态页面语言）+ tomcat（severlet 容器，提供 jsp 服务）

PHP

- B/S: php + Laravel/CakePHP

在此之中，个人建议大家使用 python，java 开发，原因无非是从学习成本，编码难度和文档丰富程度等角度出发。值得一提的是，QT 库（包括 pyQt）支持拖拽界面开发，这能让大家设计图形界面更加快速。

基本开发流程

C/S 架构：

- **根据软件的功能需求设计图形界面**。每种图形界面开发库中，都会提供一系列的图形组件，例如菜单，按钮，文本框，画布等组件，在一个主窗口中添加需要的组件，便构成了一个初步的图形界面。
- **设计针对每个交互事件的响应函数（handler），并完成绑定**。图形界面除了可以提供文本输入和显示功能之外，通常针对每个组件还有鼠标悬浮，点击，选中等事件响应机制。当我们执行这些触发动作时，就会触发一个由我们设计的响应函数来执行。因此我们可以实现用户动作到后端控制代码的映射，完成前后端的交流。
- **设计主窗口启动代码**。主窗口是应用启动的状态，也是用户使用的入口。一旦我们启动主窗口，便会打开一个客户端图形界面，该图形界面便会用绑定好的响应函数来响应的动作。

B/S 架构：

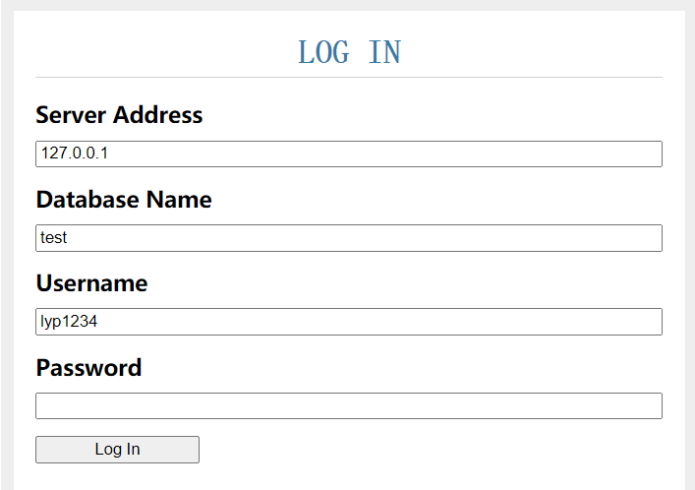
- **根据软件的功能需求设计动态网页模板**（jsp 网页，flask 中采用的 Jinja 模板库）。该模板与网页的区别是，网页是可以直接在浏览器上渲染并查看的，模板是网页之母，它用来生成一份专属网页的模板网页，并由服务器转交给用户浏览器。
- **定义用户在网页上交互事件的处理函数**，该函数能根据用户提供的参数生成动态网页并返回。每当用户在已有网页上点击某个按钮或者选中某个选项时，都可能触发一个新的网页页面请求。根据用户的某些输入值，这个网页可能是独一无二的，不可能像静态网页一样预先生成好一份。为了实时的生成这份专属网页，后端需要有一段逻辑生成该网页。通常，这些动态网页可以按照一份模板生成，后端逻辑只需要获取生成该份网页的数据，生成网页并返回该网页的工作转交给框架本身。
- **设计一个静态网页入口**。该入口作为用户从浏览器访问软件的入口。
- **错误处理的处理**。web 开发比 C/S 架构开发更容易遇到错误，例如访问页面不存在，用户登录失败等问题，进行设计时需要有显式的应对方案。

简单示例

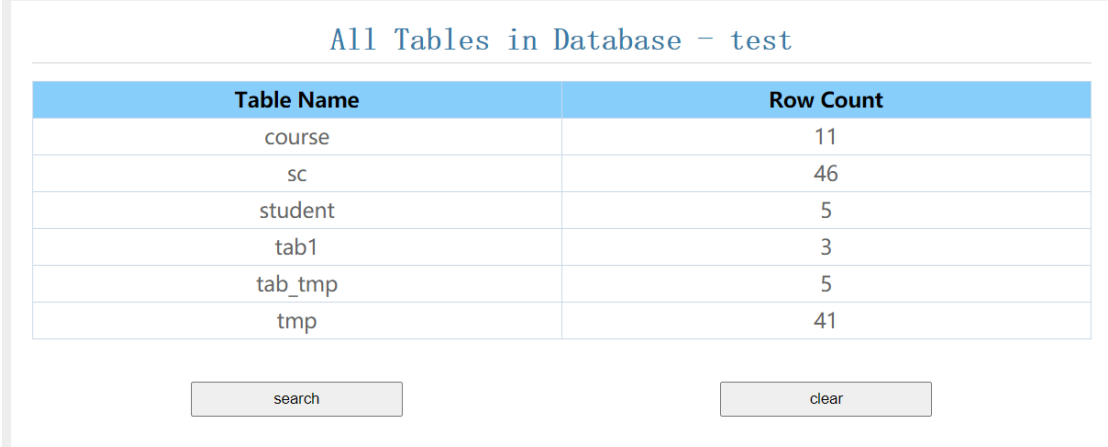
为了让软件开发不那么拿手的同学能够快速上手，我们决定在两种架构下的分别实现一个简单的 demo，并将代码开放给同学们。有需要的同学可以先弄明白 demo 的实现原理，并在上面进行功能扩展即可。

Demo 的功能为：

- 用户登录位于某个主机上的数据库



- 登录成功后展示数据库中的所有表格及表格的行数。



C/S 架构：

- python + pyQt 示例

B/S 架构：

- python + Flask 示例
- java + jsp + tomcat