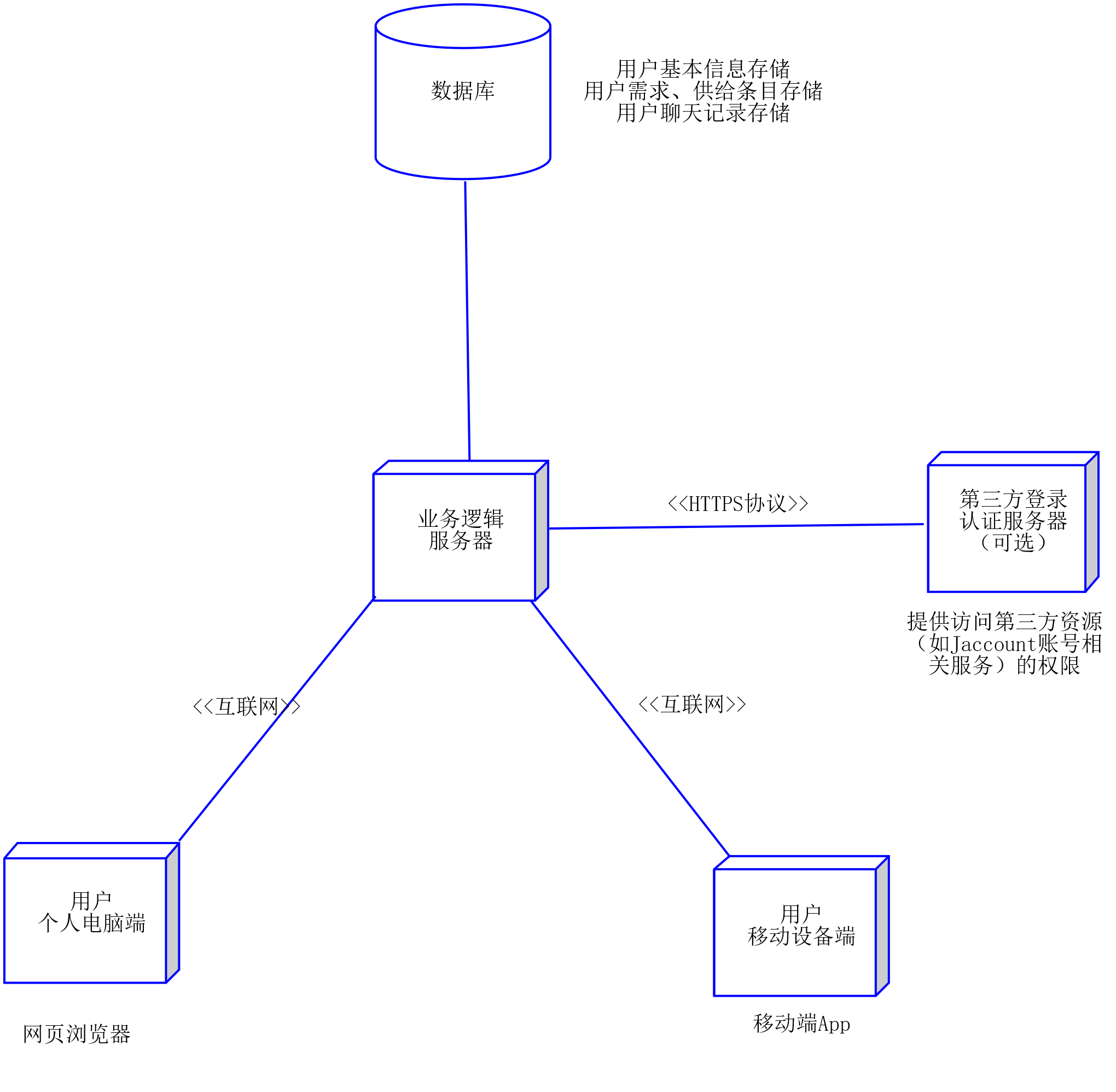
### 部署（物理）视图



此软件部署上采用B/S架构。

用户设备不做任何业务逻辑的处理。用户端基于HTTPS协议，通过互联网访问服务器主机（移动设备APP基于网页浏览器自动打包生成）。此外，聊天功能基于WebSocket协议，由服务器主机负责转发消息，考虑加入消息中间件（optional）

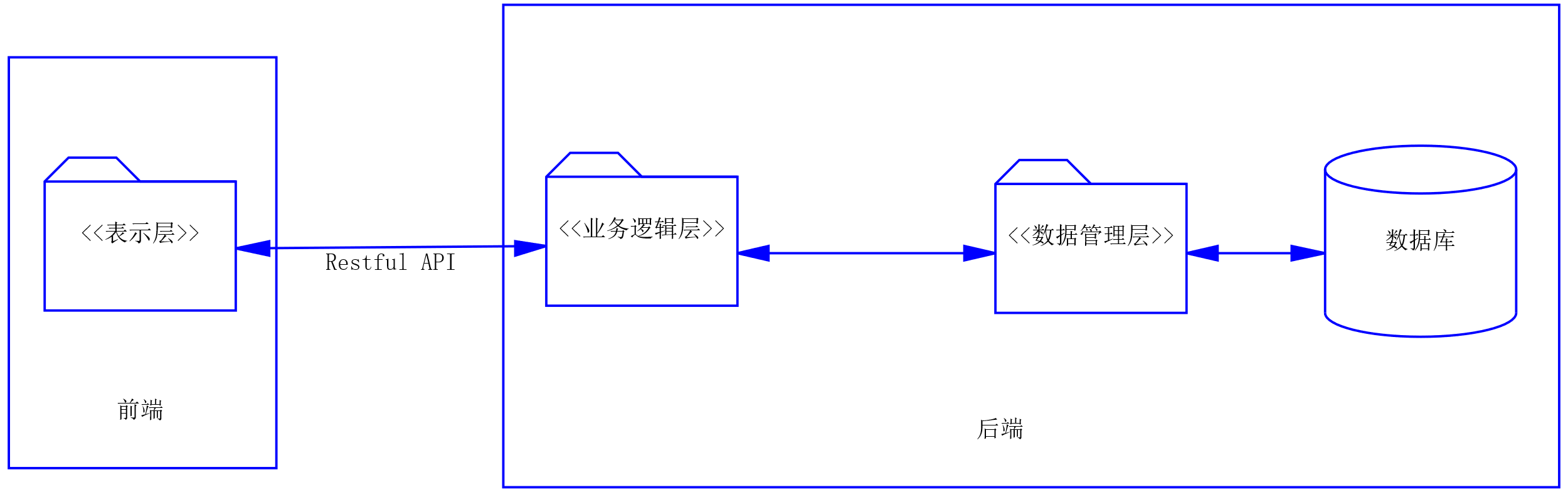
服务器端处理客户端请求并作出响应，实现软件全部功能：用户登录、需求管理、需求推送流访问与查询、用户聊天等。

其中，认证服务器是一个可选项，具体取决于软件用户登录的实现是否需要第三方认证（如JACCOUNT认证）。

数据库在设计中下与服务器部署于同一主机，由于可以在部署上实现解耦，所以在视图中标出为两个分离的部分。

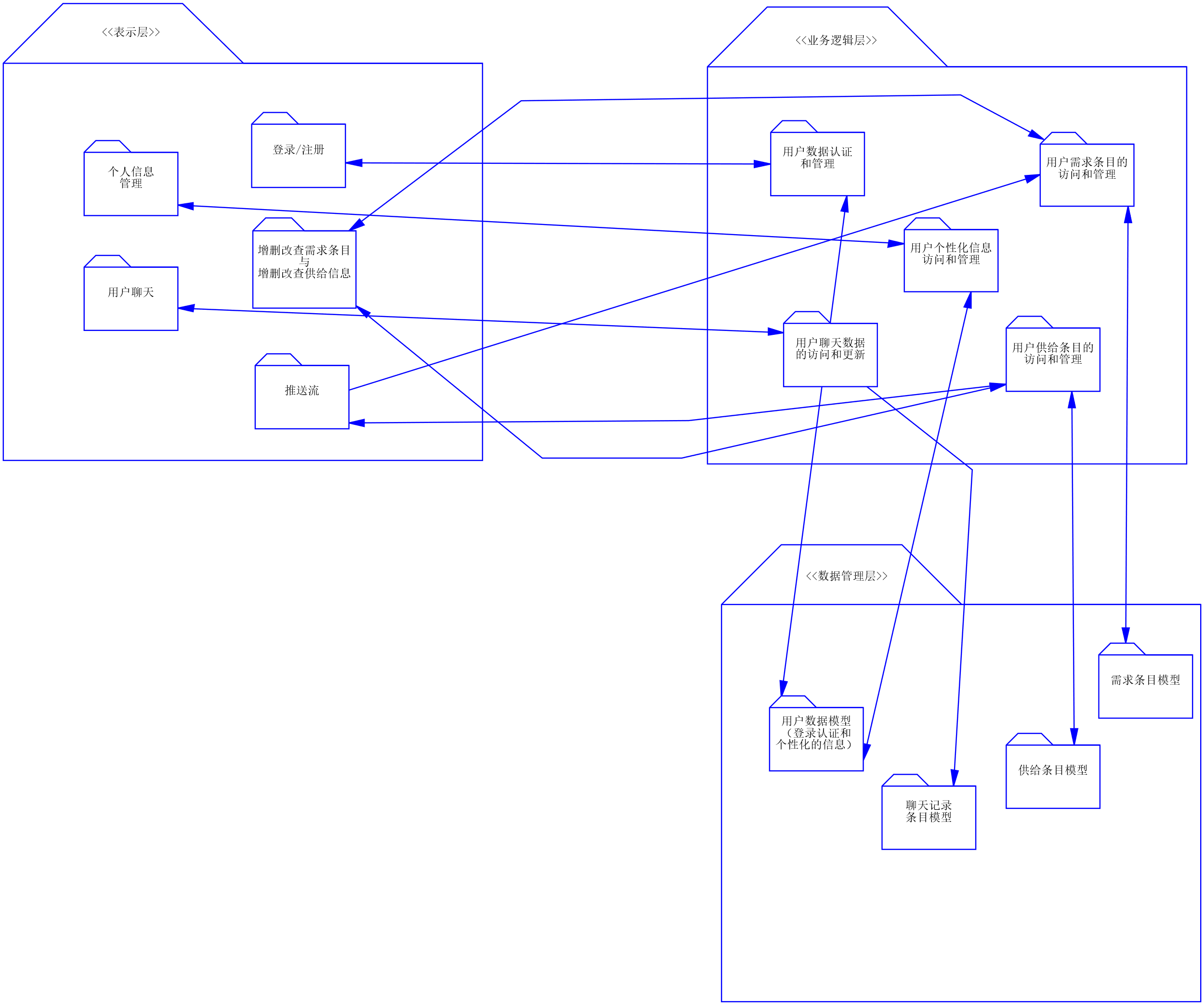
### 逻辑视图

大体上的逻辑视图如下：



此软件采用前后端分离的设计风格，后端使用Django开发默认的MVC设计风格（准确地说是MTV模式，并且Template模块由分离出的前端负责，后端只负责处理请求，发送数据）。整体上看，可以认为是3-Tier逻辑架构。

具体的细节如下：



其中，前端表示层处理页面逻辑。管理向用户展示的内容，响应用户操作，并处理诸页面之间的跳转逻辑关系。表示层根据对应的需求，通过Restful API（实现中采用Django Rest Framework）向业务逻辑层发送请求，并获得Json数据响应以渲染页面。

后端业务逻辑层处理前端传来的需求。根据对应的任务调用数据管理层对应模型以访问、更新或是删除数据库中的对应表单项。

数据处理层定义了一些模型，对应Django MTV架构中的Model，提供了一种方便的面向对象的管理数据库的接口。一个数据表被抽象为一个Model类，而一个表单项则被抽象为对应的类对象。