**广州应用科技学院**

**计算机科学与技术系**

**题 目： 观影通**

**专业班级： 20计科10班**

**学 号： 2008011050**

**学生姓名： 陈秋丽**

**授课教师：** **潘伯新老师**

**2023年 5月 29 日**

目录

**[一、 需求分析 3](#_Toc19448)**

[1.总体需求分析 3](#_Toc26610)

[2.功能分析 3](#_Toc22799)

[3.性能分析 4](#_Toc2824)

[3.3.1登录用例规约表 6](#_Toc10203)

[3.3.2浏览电影信息用例规约表 7](#_Toc12570)

[3.3.3管理个人信息用例规约表 8](#_Toc20068)

[3.4系统类图 9](#_Toc20068)

[3.5顺序图 9](#_Toc20068)

[3.5.1 登录顺序图 9](#_Toc737)

[3.5.2 浏览电影信息顺序图 10](#_Toc23052)

[3.5.3 管理个人信息顺序图 11](#_Toc27035)

[3.6状态图 12](#_Toc25326)

[3.7活动图 13](#_Toc2405)

[3.8 部署图 13](#_Toc32607)

**[二、 数据库设计 14](#_Toc16197)**

[1. 总体设计 14](#_Toc16455)

[2. 模块设计 15](#_Toc18578)

[3. 高可用、容灾方案设计 34](#_Toc25963)

[4. 高并发高负载扩展方案设计 35](#_Toc9891)

[5. 安全设计 36](#_Toc12512)

1. **需求分析**

1.总体需求分析

观影通是一款在线电影票务平台，用户可以通过观影通网站或手机应用浏览电影信息、影院信息、场次信息，切换城市浏览该城市的电影信息。用户可以注册观影通，成为用户，可以修改自己的个人信息。

2.功能分析

观影通根据用户对象主要分成三部分功能模块，分别为游客功能模块、用户功能模块和通用功能模块。其基本功能模块如图1-1所示，具体描述如下：

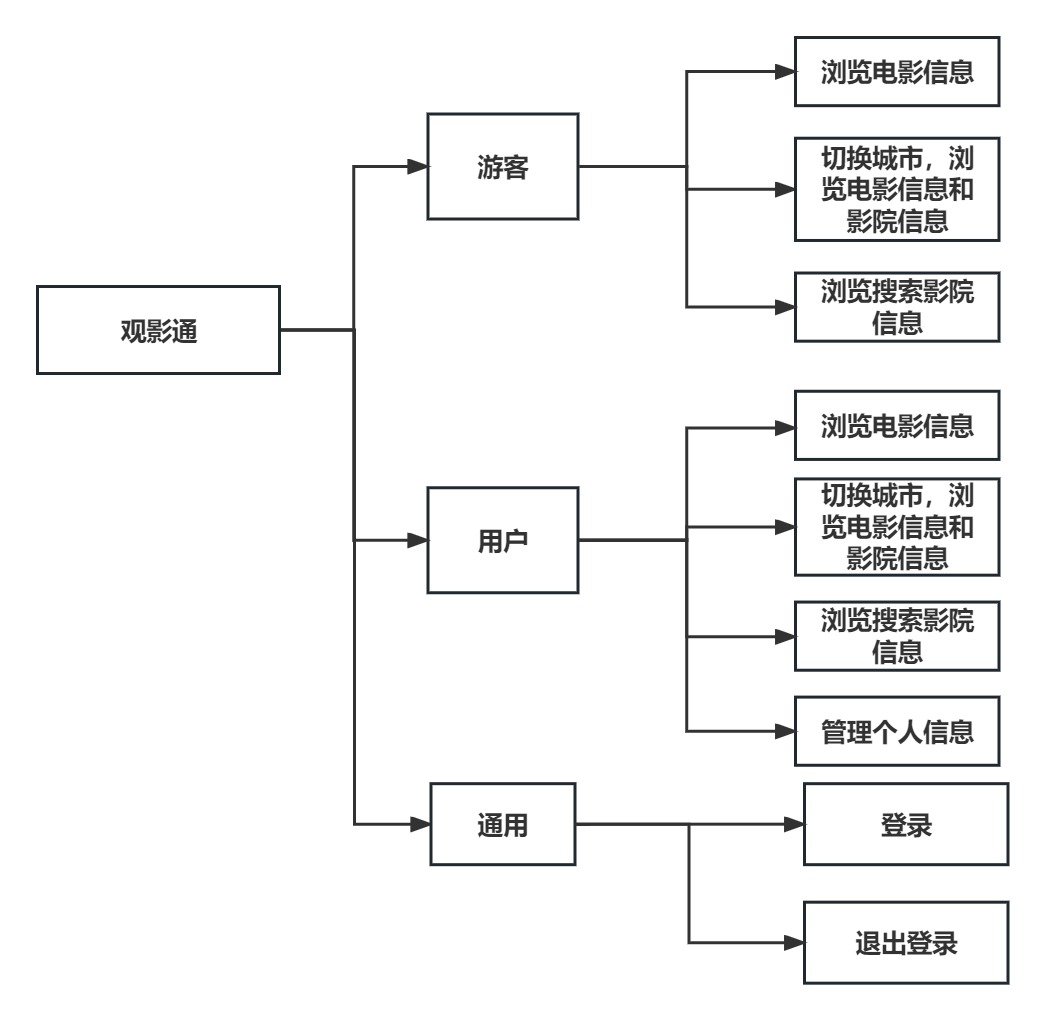


图1-1 观影通功能模块

2.1游客功能模块

（1）浏览电影信息

游客浏览电影信息，包括正在热映和即将上映的电影，点击电影信息查看具体的电影详情。

（2）切换城市，浏览电影信息和影院信息

游客切换城市，展示不同城市的电影信息和影院信息。

（3）浏览搜索影院信息

游客浏览影院信息，搜索具体影院进入影院信息页面选座。

2.2用户功能模块

（1）浏览电影信息

用户浏览电影信息，包括正在热映和即将上映的电影，点击电影信息查看具体的电影详情。

（2）切换城市，浏览电影信息和影院信息

用户切换城市，展示不同城市的电影信息和影院信息。

（3）浏览搜索影院信息

用户浏览影院信息，搜索具体影院进入影院信息页面选座。

（4）管理个人信息

用户登录后，进入我的页面，可以对个人信息进行修改。

2.3通用功能模块

（1）登录

（2）退出登录

3.性能分析

3.1系统用例图

用例图（use case diagram）是用户与系统交互的最简表示形式，展示用户和与之相关的用例之间的关系。用例是从用户的观点对系统行为的一个描述。对于系统开发人员而言，用例是一个有价值的工具，是用来从用户的观察角度收集系统需求的一项技术。用例图用于说明如何使用系统以及系统的需求，并表达用例与角色之间的关系。该系统的用例图如图1-2所示：

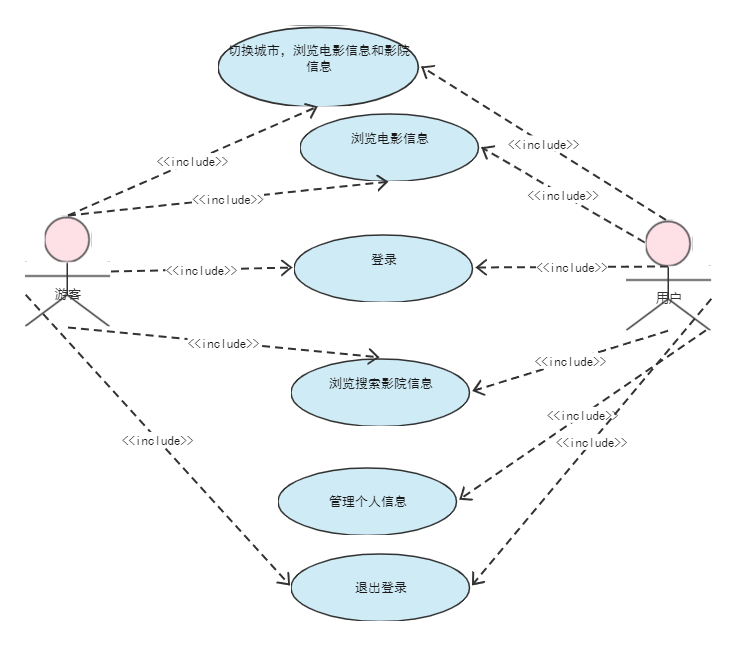


图1-2 系统用例图

3.2系统详细用例

表1-1 系统的详细用例

|  |  |  |
| --- | --- | --- |
| 参与者 | 特性 | 用例 |
| 游客 | 游客进入系统，点击电影页面，查看正在热映和即将上映的电影，点击某一个电影查看电影详情。 | 浏览电影信息 |
| 游客进入系统，切换城市，可以查看不同城市的电影信息和影院信息。 | 切换城市 |
| 游客进入系统，点击影院页面，查看所有影院信息，并支持搜索，点击影院进入影院详情。 | 浏览影院信息 |
| 游客点击我的页面，系统拦截跳转登录页进行登录。 | 登录 |
| 游客登录成功后，退出登录，退出系统。 | 退出登录 |
| 用户 | 用户进入系统，点击电影页面，查看正在热映和即将上映的电影。 | 浏览电影信息 |
| 用户进入系统，切换城市，可以查看不同城市的电影信息和影院信息。 | 切换城市 |
| 用户进入系统，点击影院页面，查看所有影院信息，并支持搜索，点击影院进入影院详情。 | 浏览影院信息 |
| 用户点击我的页面，系统拦截跳转登录页进行登录。 | 登录 |
| 用户登录成功后，退出登录，退出系统。 | 退出登录 |
| 用户登录后，进入我的页面，编辑个人资料：上传头像和修改昵称、性别和生日。 | 管理个人信息 |

3.3用例规约表描述

## 3.3.1登录用例规约表

表1-2登录用例规约表

|  |  |
| --- | --- |
| 用例名称 | 登录 |
| 标识符 | UC0000 |
| 用例描述 | 用户或游客利用该用例登录系统，通过身份验证后获得相应的操作权限 |
| 参与者 | 用户（1）、游客（2） |
| 优先级 | 1 |
| 状态 | 通过审查 |
| 前置条件 | 无 |
| 后置条件 | 登录成功，显示相应的需要权限的界面 |
| 基本事件流 | 1. 用户或游客需要登录该系统； 2. 系统显示登录页面，并要求输入手机号和验证码； 3. 用户或游客输入手机号和验证码； 4. 系统验证该手机号的合法性，允许用户登录系统； 5. 登录完成后，进入页面。 |
| 备选事件流 | 1. 未输入手机号直接登录，提示输入手机号； 2. 登录的手机号不合法，提示正确输入手机号； |
| 被泛化的用例 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 修改历史记录 |  |

## 3.3.2浏览电影信息用例规约表

表1-3 浏览电影信息用例规约表

|  |  |
| --- | --- |
| 用例名称 | 浏览电影信息 |
| 标识符 | UC0001 |
| 用例描述 | 用户或游客可通过该用例浏览系统的所有电影信息 |
| 参与者 | 用户（1）、游客（2） |
| 优先级 | 1 |
| 状态 | 通过审查 |
| 前置条件 | 无 |
| 后置条件 | 用户或游客正常查询到电影信息 |
| 基本事件流 | 1. 用户或游客进入系统，点击电影页面； 2. 系统显示默认地址的所有电影信息； 3. 用户或游客查看到正在热映和即将上映的电影； 4. 用户或游客点击某一电影进入电影详情页，查看到更多的信息。 |
| 备选事件流 | 无 |
| 被泛化的用例 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 修改历史记录 |  |

## 3.3.3管理个人信息用例规约表

表1-4管理个人信息用例规约表

|  |  |
| --- | --- |
| 用例名称 | 管理个人信息 |
| 标识符 | UC0002 |
| 用例描述 | 用户登录该系统，可以编辑个人信息。 |
| 参与者 | 用户 |
| 优先级 | 无 |
| 状态 | 通过审查 |
| 前置条件 | 用户已登录系统。 |
| 后置条件 | 成功进入我的页面。 |
| 基本事件流 | 1. 用户进入我的页面； 2. 点击头像上传，或者点击我的姓名进入个人资料页，编辑个人信息并保存； 3. 我的页面显示编辑成功的个人信息，展示头像和姓名。 |
| 备选事件流 | 无 |
| 被泛化的用例 | 无 |
| 被包含的用例 | 无 |
| 被扩展的用例 | 无 |
| 修改历史记录 |  |

3.4 系统类图

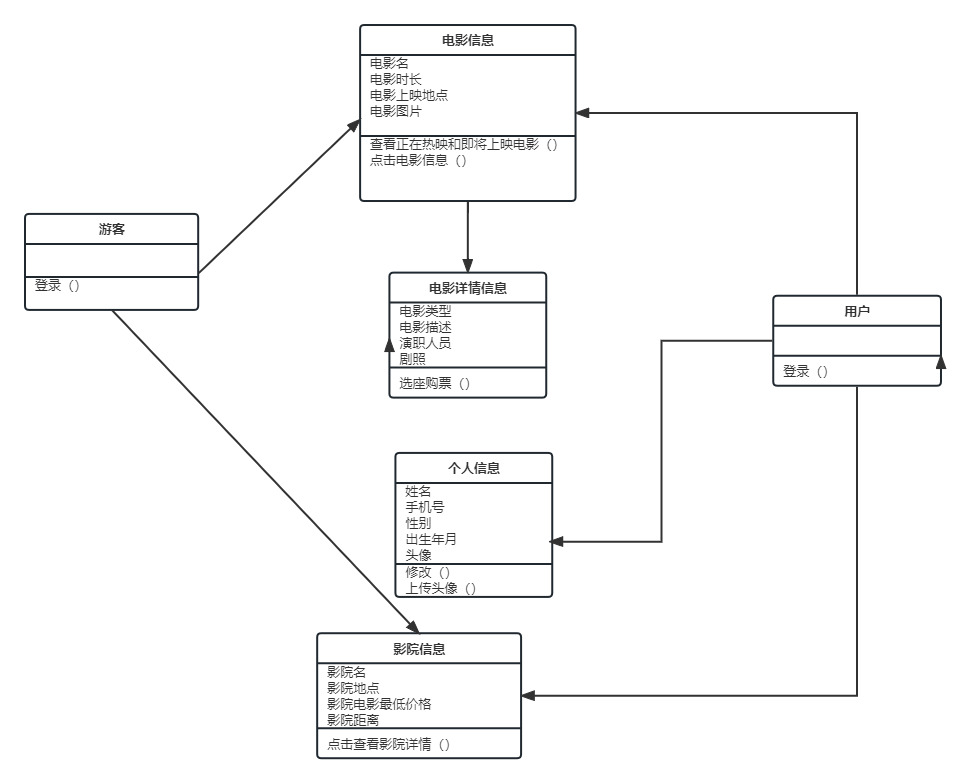


图1-3 系统类图

3.5 顺序图

## 3.5.1 登录顺序图

① 游客或用户进入我的页面，系统跳转登录页

② 系统显示登录页面

③ 游客或用户输入手机号登录

④ 系统校验手机号非空和合法性

⑤ 点击登录与数据库比对，存在该手机号直接登录；没有该手机号新增完成登录

⑥ 游客或用户成功进入我的页面

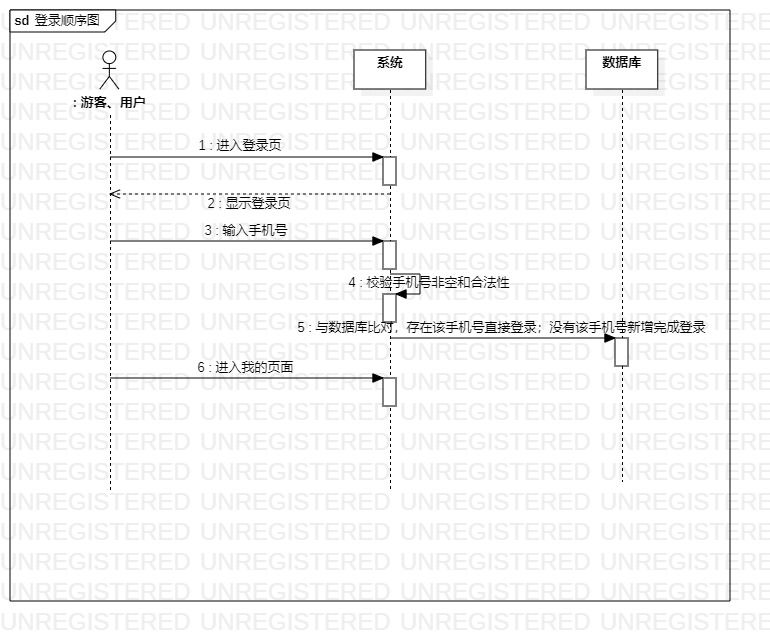


图1-4 登录顺序图

## 3.5.2 浏览电影信息顺序图

① 游客或用户进入系统，点击电影页面

② 系统显示显示所有正在热映和即将上映的电影信息

③ 点击电影跳转进入详情页

④ 系统显示电影详情信息

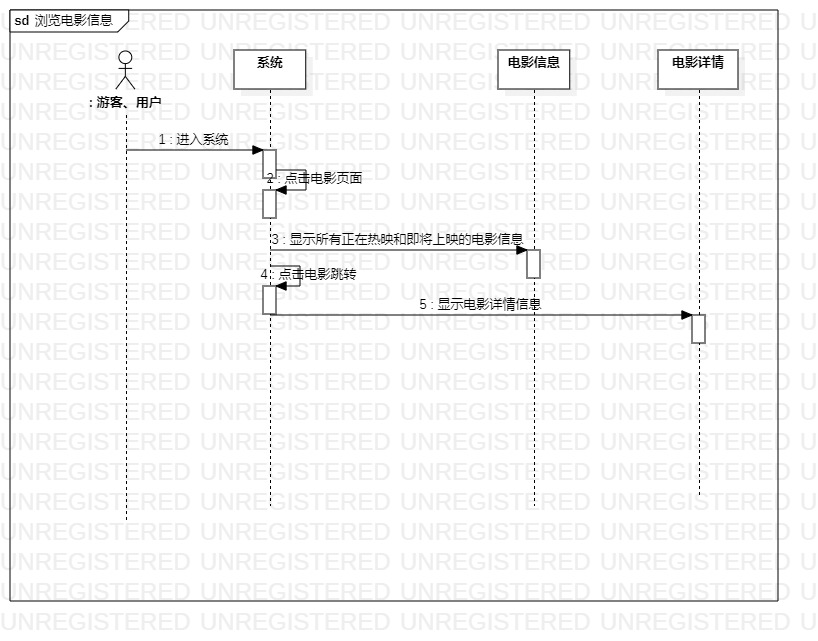


图1-5 浏览电影信息顺序图

## 3.5.3 管理个人信息顺序图

① 用户进入系统，点击我的页面

② 点击头像上传更换头像；或点击姓名进入个人资料页

③ 编辑头像，昵称，性别和出生

④ 点击保存，更新进数据库

⑤ 系统显示修改信息成功

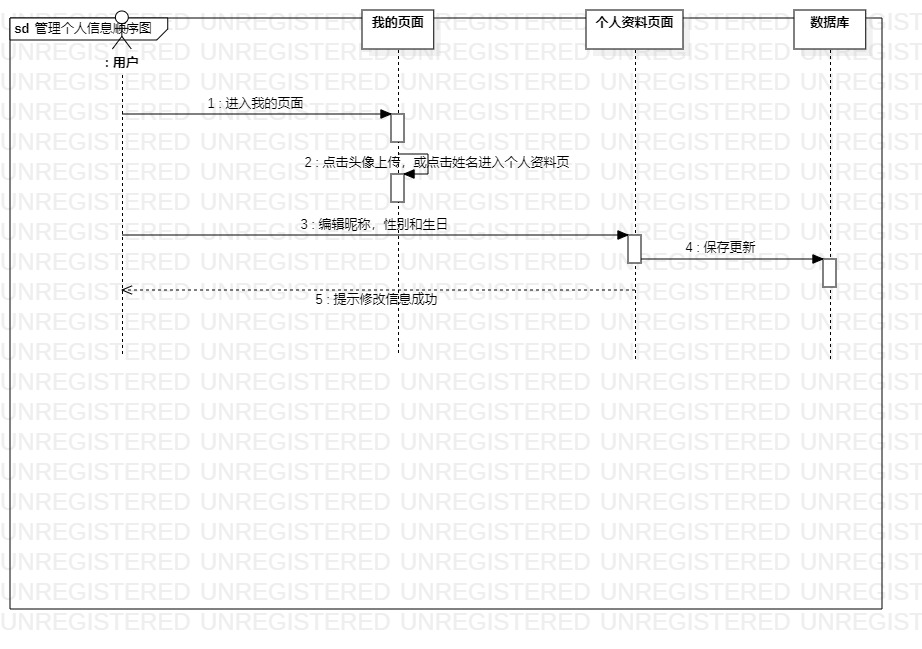


图1-6 管理个人信息顺序图

## 3.6状态图

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 待售 | 预订中 | 已售 | 已出票 | 退票中 | 已退票 | 作废 |
| 待售 | -- | 用户选定了场次和座位号，等待支付 | -- | -- | -- | -- | -- |
| 预订中 | -- | -- | 支付成功 | -- | -- | -- | -- |
| 已售 | -- | -- | -- | 配票成功 | -- | -- | -- |
| 已出票 | -- | -- | -- | -- | 发起退票请求 | -- | -- |
| 退票中 | -- | -- | -- | -- | -- | 返还金额 | -- |
| 已退票 | -- | -- | -- | -- | -- | -- | 从订单列表删除 |
| 作废 | -- | -- | -- | -- | -- | -- | -- |



图1-7 电影票状态图

## 3.7活动图

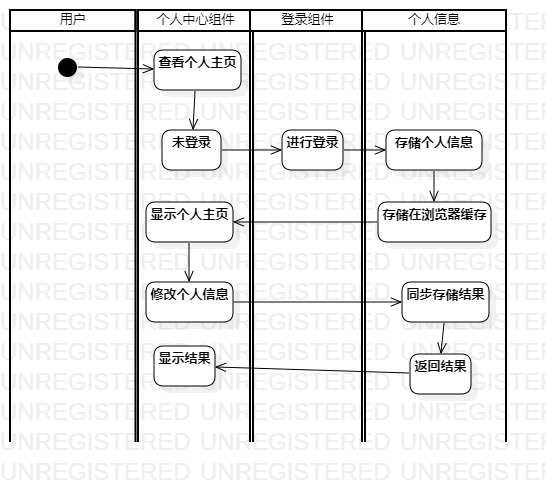


图1-8 修改个人信息活动图

## 3.8 部署图

1. **数据库设计**
2. 总体设计

该项目使用MongoDB数据库，MongoDB是一种面向文档的NoSQL数据库管理系统，它具有许多优势：

①灵活的数据模型：MongoDB使用文档模型来存储数据，文档是以键值对的形式组织的，可以嵌套和包含各种数据类型。这种灵活性使得MongoDB适用于处理各种类型的数据，并且可以轻松地进行模式更改和演化。

②高扩展性：MongoDB可以轻松地扩展以应对不断增长的数据量和流量。它支持水平扩展，可以通过添加更多的服务器节点来增加存储容量和处理能力。这种可扩展性使得MongoDB非常适合处理大规模数据和高流量应用程序。

③高性能：MongoDB具有快速的读写操作能力。它使用了内存映射和索引等技术来提供高效的数据访问。此外，MongoDB还支持复制和故障转移机制，以确保数据的高可用性和可靠性。

根据需求分析和系统分析，规划本系统的数据库实体共有4个，分别为影院实体、城市实体、用户实体、电影实体。

（1）各个实体拥有的属性如下：

1）电影院实体的属性有影院Id、影院名、影院地址、城市Id等。每一个影院有唯一的影院Id。

2）城市实体的属性有城市Id、城市名。每一个城市有唯一的城市Id。

3）用户实体的属性有用户姓名、性别、手机号、头像、生日。每一个用户有唯一的手机号。

4）电影实体的属性有电影Id、电影名、电影宣传照、电影评分、城市Id等。每一部电影有唯一的电影Id。

（2）实体间约束如下：

1）一个城市可以有多个电影院，一个影院可以有多个电影。

（3）将上述描述转换成关系模式如下（下划直线表示主键，下划曲线表示外键，下划虚线表示既是主键也是外键）：

影院关系模式：（影院Id、影院名、影院地址、城市Id ）

城市关系模式：（城市Id、城市名）

用户关系模式：（手机号、用户姓名、性别、头像、生日）

电影关系模式：（电影Id 、电影名、电影宣传照、电影评分、城市Id）

1. 模块设计

观影通系统，前端使用React库，后端使用nodejs的Express框架，数据库使用MongoDB。后端服务封装连接MongoDB数据库，代码如图2-1所示：

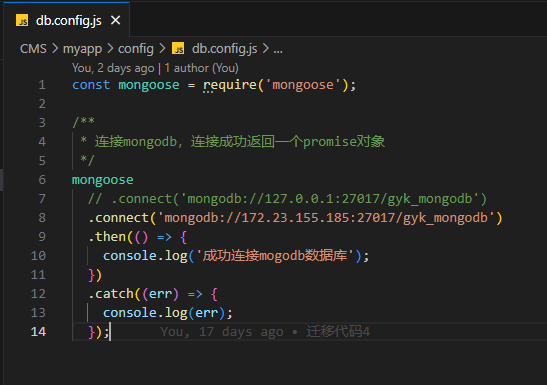


图2-1 后端连接MongoDB数据库

（1）电影模块：

在MongoDB命令窗口导入json数据，json数据源如图2-2所示，导入数据库后视图界面如图2-3所示：

mongoimport --host localhost --port 27017 --db gyk\_mongodb --collection films --file "F:\gky\gky-mongodb\大作业\DB\dbdata\film.json" --jsonArray

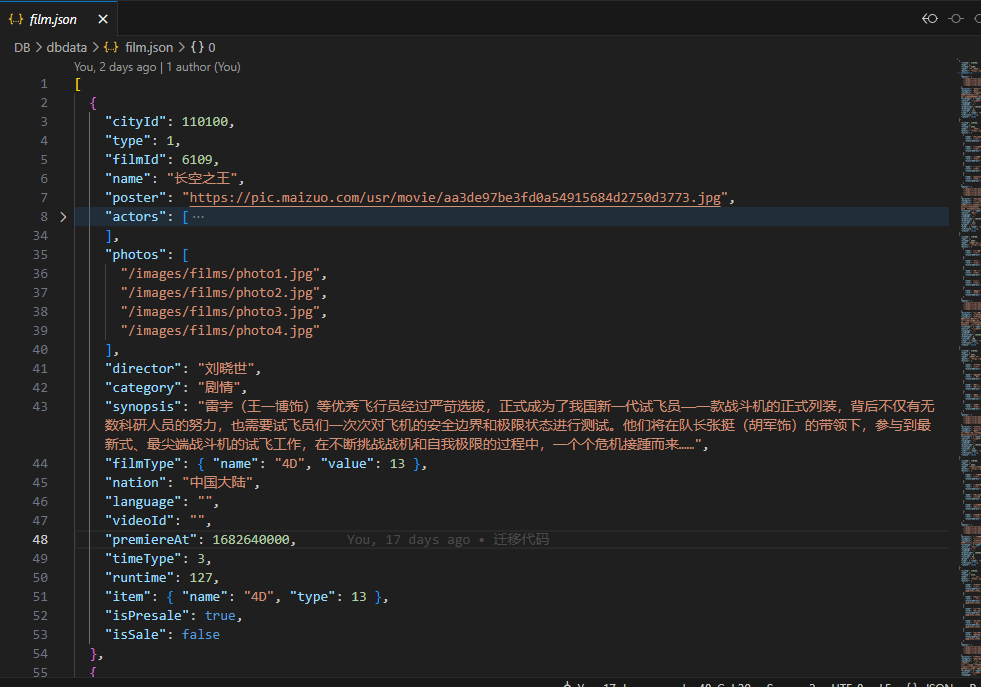


图2-2 Films集合json数据

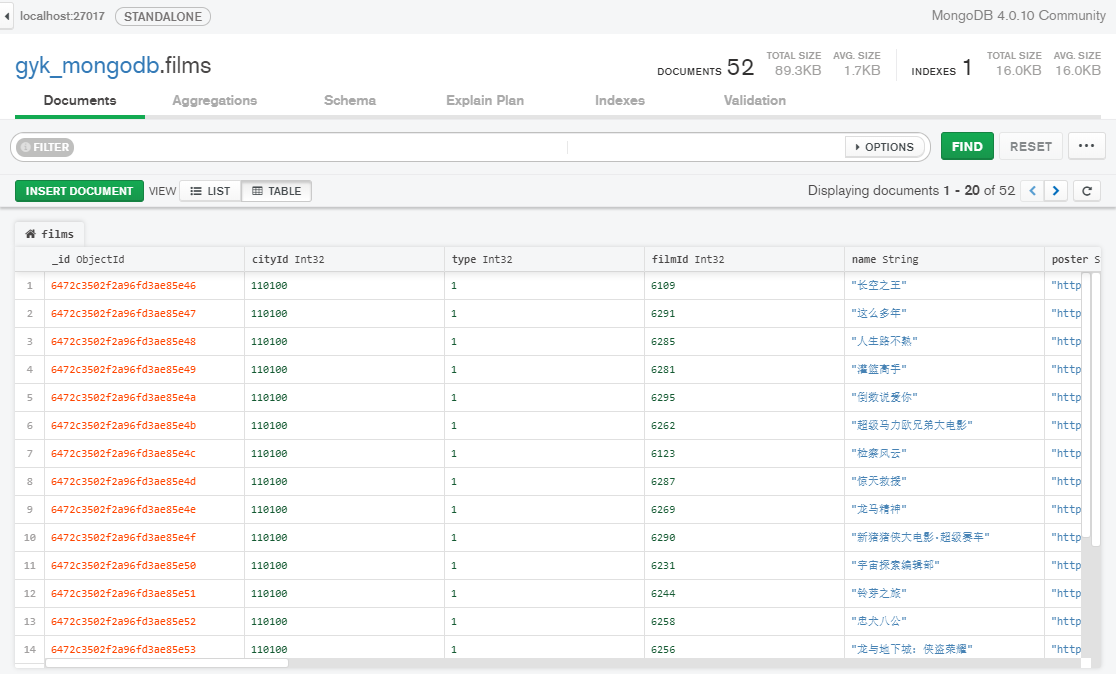


图2-3 FIlms集合视图

后端服务使用Mongoose创建FilmsModel模型，代码如图2-4所示。接口设计开发根据FilmsModel模型提供，代码如图2-5所示：

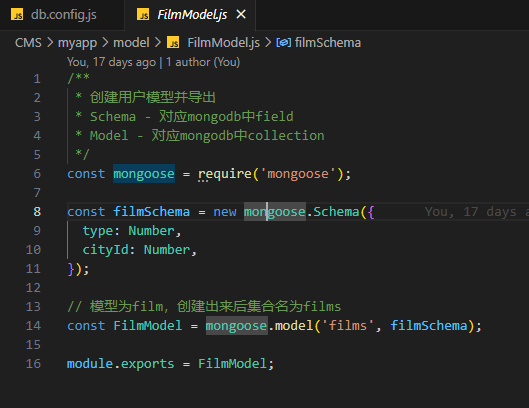


图2-4 FIlmsModel模型



图2-3 FIlms模块接口设计

电影模块列表界面和电影模块详情界面如图2-5和2-6所示：



图2-5 电影模块列表界面



图2-6 电影模块详情界面

（2）影院模块：

在MongoDB命令窗口导入json数据，json数据源如图2-2所示，导入数据库后视图界面如图2-3所示：

mongoimport --host localhost --port 27017 --db gyk\_mongodb --collection cinmeas --file "F:\gky\gky-mongodb\大作业\DB\dbdata\cinema.json" --jsonArray

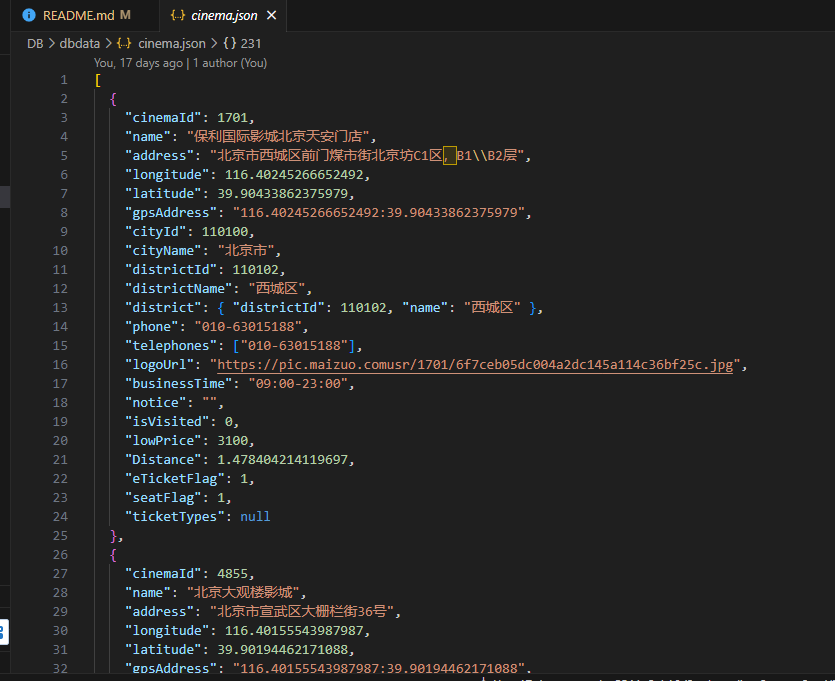


图2-7 Cinemas集合json数据

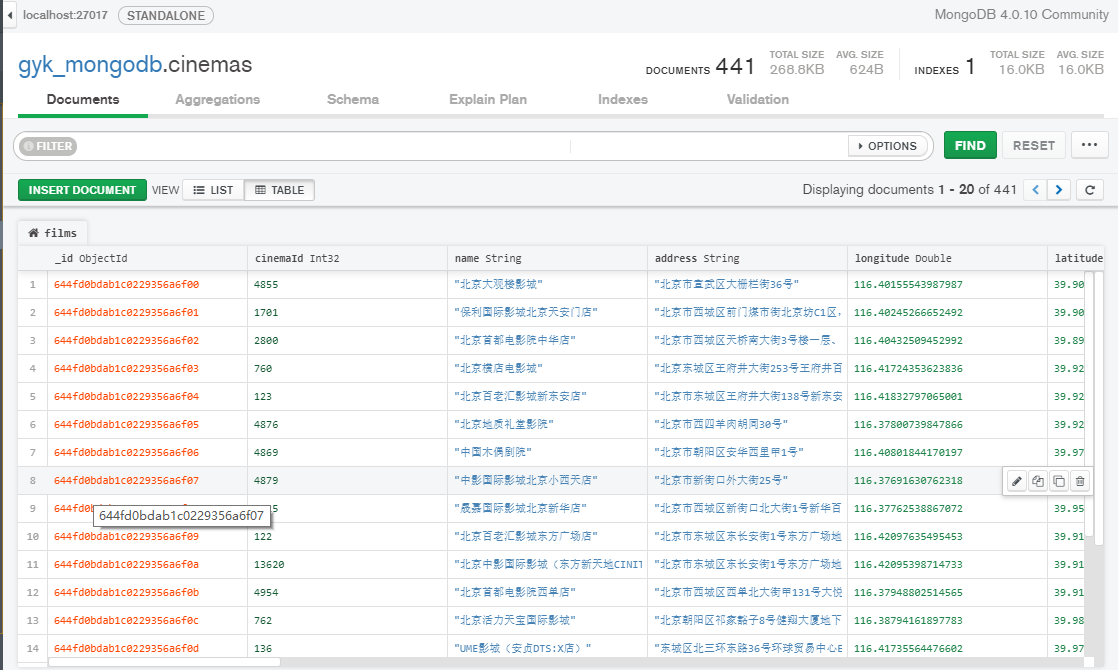


图2-8 Cinemas集合视图

后端服务使用Mongoose创建CinemasModel模型，代码如图2-9所示。接口设计开发根据CinemasModel模型提供，代码如图2-10所示：

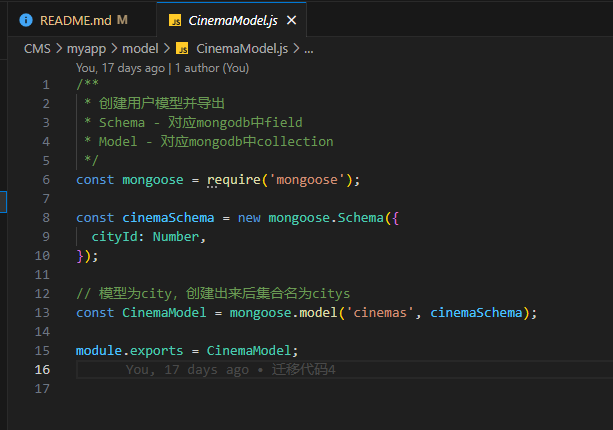


图2-9 CinemasModel模型

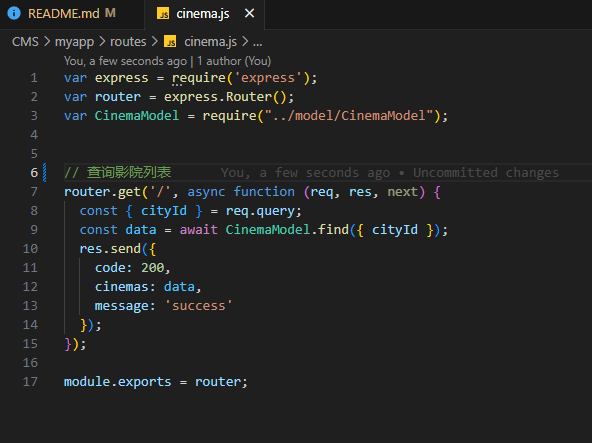


图2-10 Cinemas模块接口设计

影院模块列表界面如图2-11所示：



图2-11 影院模块列表界面

（3）个人中心模块：

在MongoDB命令窗口导入json数据，json数据源如图2-12所示，导入数据库后视图界面如图2-13所示：

mongoimport --host localhost --port 27017 --db gyk\_mongodb --collection users --file "F:\gky\gky-mongodb\大作业\DB\dbdata\user.json"

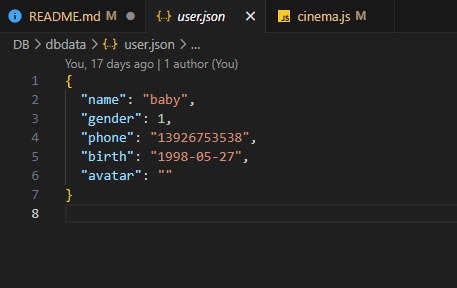


图2-12 Userss集合json数据

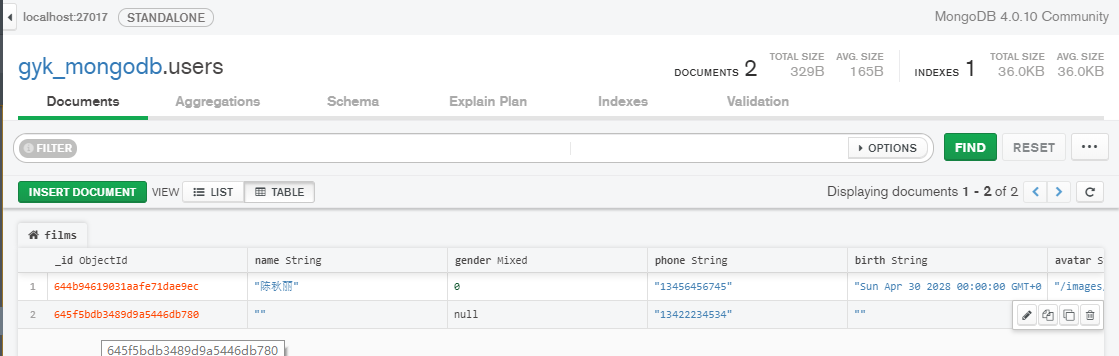


图2-13 Users集合视图

后端服务使用Mongoose创建UsersModel模型，代码如图2-14所示。接口设计开发根据UsersModel模型提供，代码如图2-15所示：

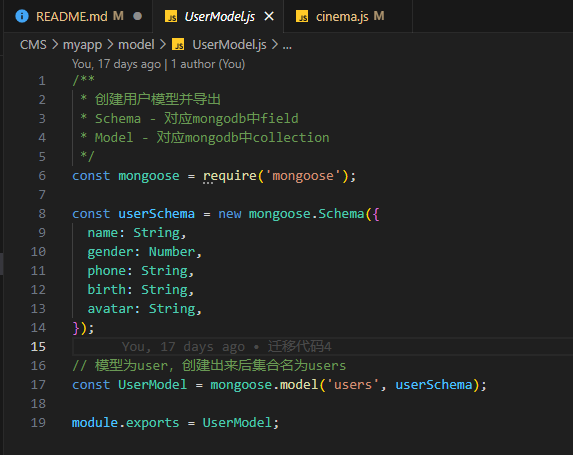


图2-16 UsersModel模型

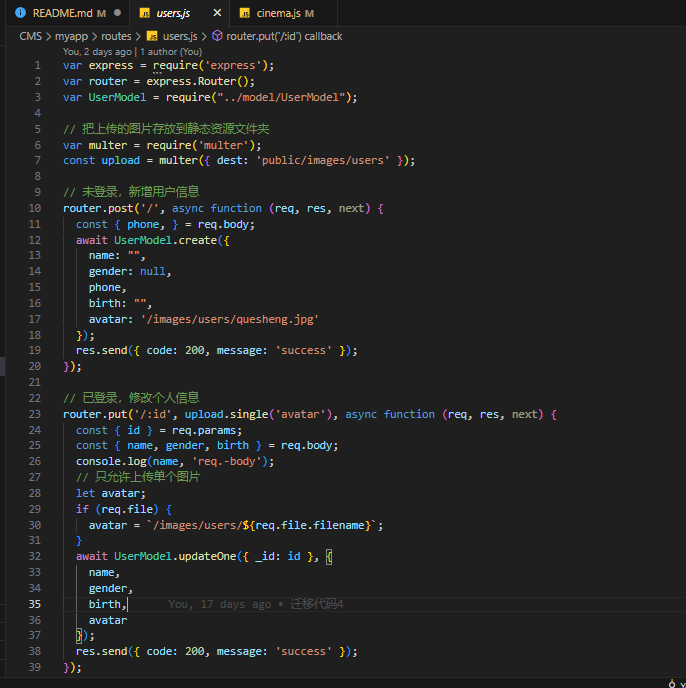


图2-17 Users模块接口设计

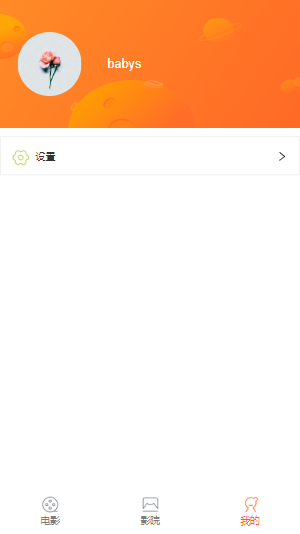
个人中心展示界面做了路由拦截，如果未登录状态跳转到登录页，已登录展示个人信息，如图2-18所示。个人信息修改页面如图2-19至2-21所示 ：

图2-18 个人中心模块界面



图2-19 个人中心修改界面



图2-20 个人中心修改成功界面

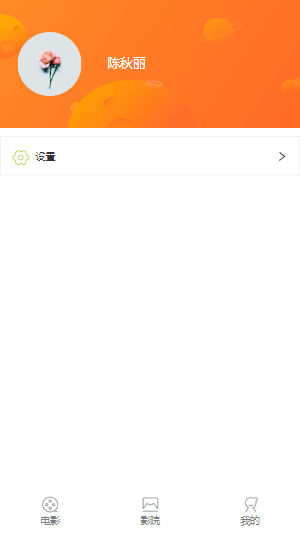


图2-21 个人中心修改成功后展示个人信息界面

（4）城市模块：

在MongoDB命令窗口导入json数据，json数据源如图2-22所示，导入数据库后视图界面如图2-23所示：

mongoimport --host localhost --port 27017 --db gyk\_mongodb --collection citys --file "F:\gky\gky-mongodb\大作业\DB\dbdata\city.json" --jsonArray

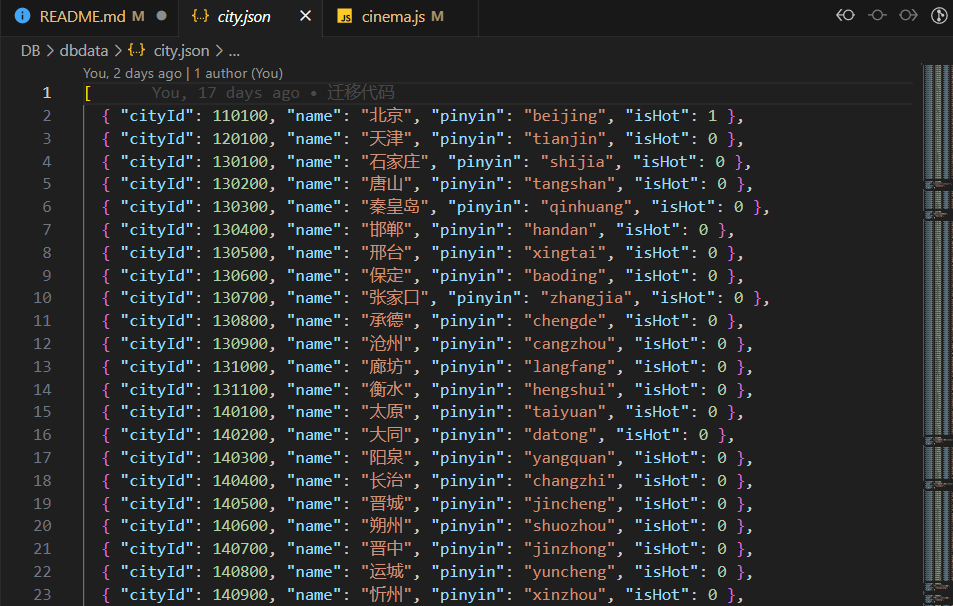


图2-22 Citys集合json数据

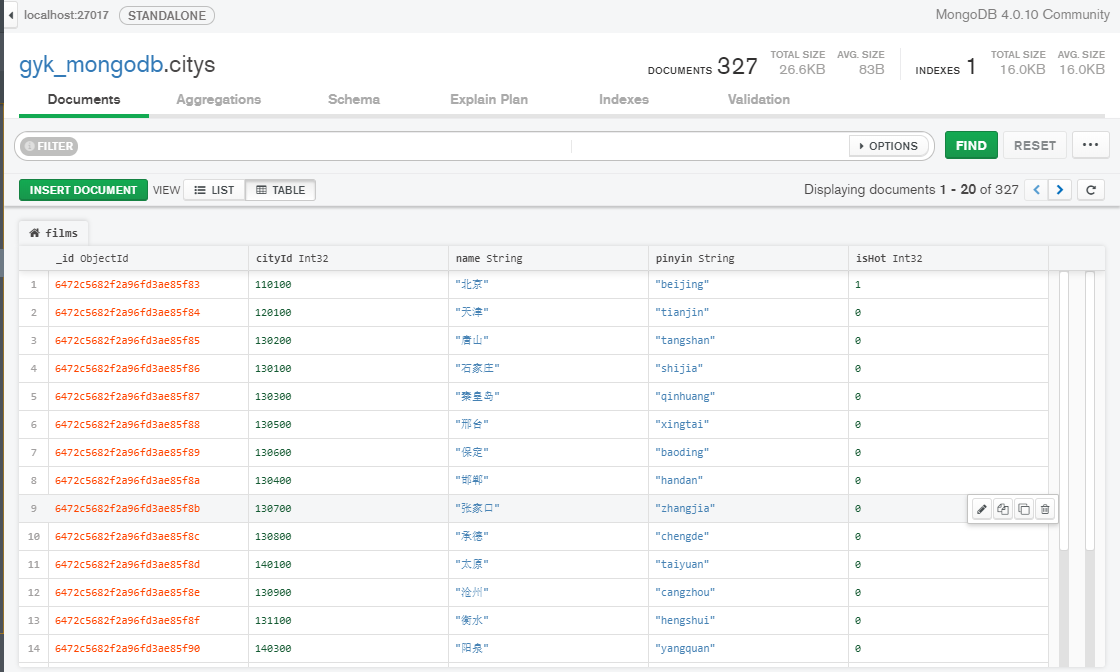


图2-23 Citys集合视图

后端服务使用Mongoose创建CitysModel模型，代码如图2-24所示。接口设计开发根据CitysModel模型提供，代码如图2-25所示：

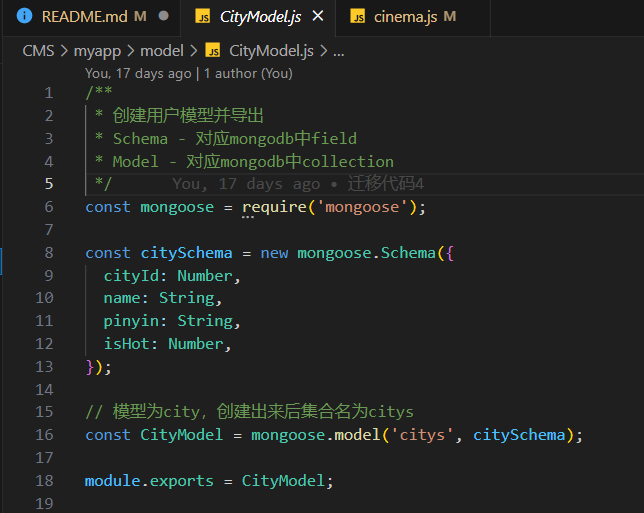


图2-24 CitysModel模型

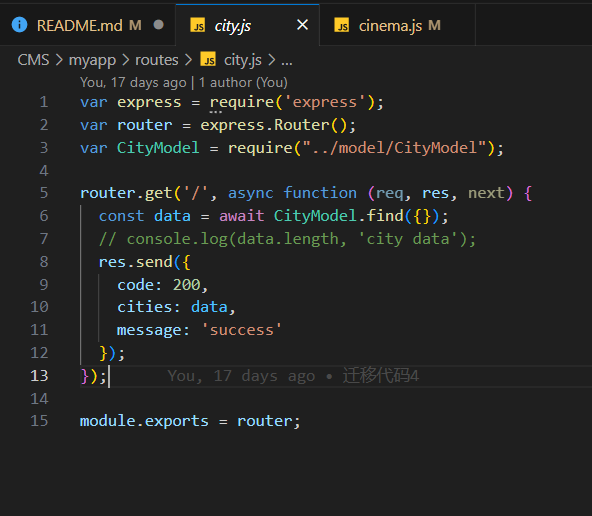


图2-25 Citys模块接口设计

城市模块界面，选择城市，切换不同的电影和影院数据，如图2-26所示：



图2-26 城市模块界面

1. 高可用、容灾方案设计

把前端、后端、数据库部署到阿里云服务器docker容器上，通过ssh连接上公网ip，显示的已部署的docker容器如下图2-27所示：

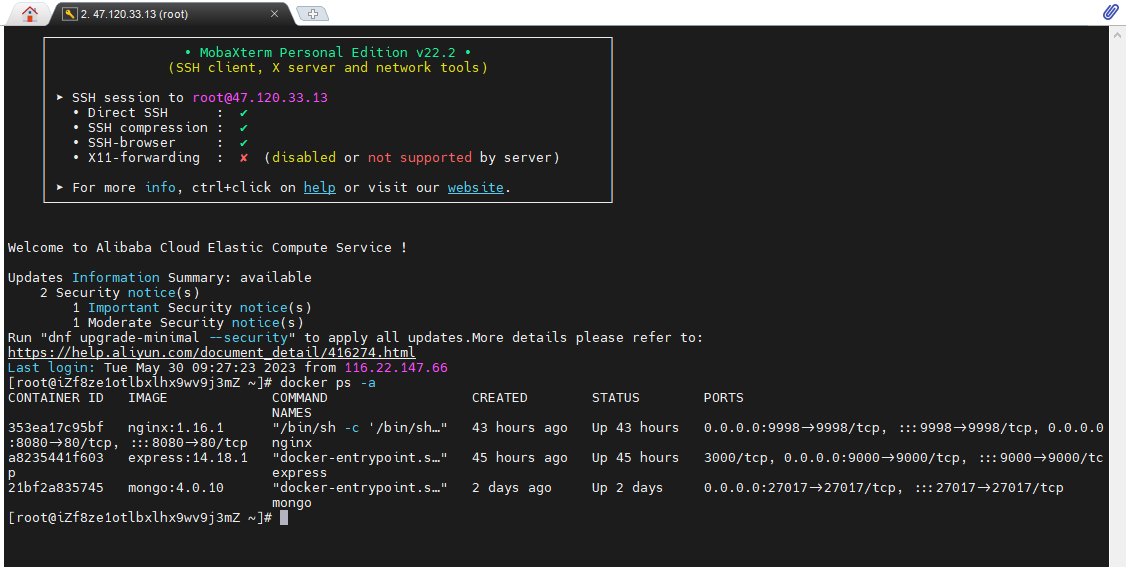


图2-27 部署的docker容器图

1. 高并发高负载扩展方案设计

在电影模块列表页面，电影信息数据库里面几十条，前端界面视图高度有限，如果把所有的数据一下子请求回来，会导致页面首屏加载过慢，用户体验不好。因此，我在后端接口对电影信息的列表数据做了分页请求，这样可以让前端按需请求，代码如图2-28所示。同时由于该app是个单页面应用程序，首次加载如果不做处理默认加载整个项目打包后的js文件，我在前端也对电影信息模块做了路由懒加载处理，拆分打包体积，提高首屏加载速度，代码如图2-29所示：

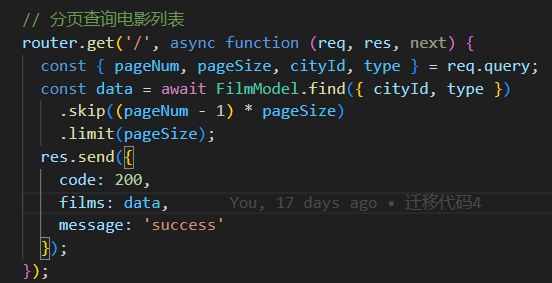


图2-28 分页代码

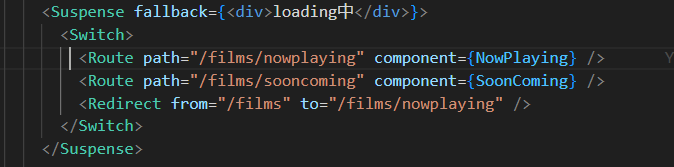


图2-29 路由懒加载代码

1. 安全设计

无