便民体检预约小程序

小组成员：2014568 范宸华　2014579 梁宏伟

小程序说明

本小程序为 2022 年微信小程序大赛参赛作品，主题为便民生活的医院体检预约小程序，本文档适用于描述迭代 4(v1.0.0)，为最终版文档。项目实现了一个包含前后端的体检预约小程序，前端使用 vue 框架，后端为 springboot 框架，建立了微信云托管。

应用场景

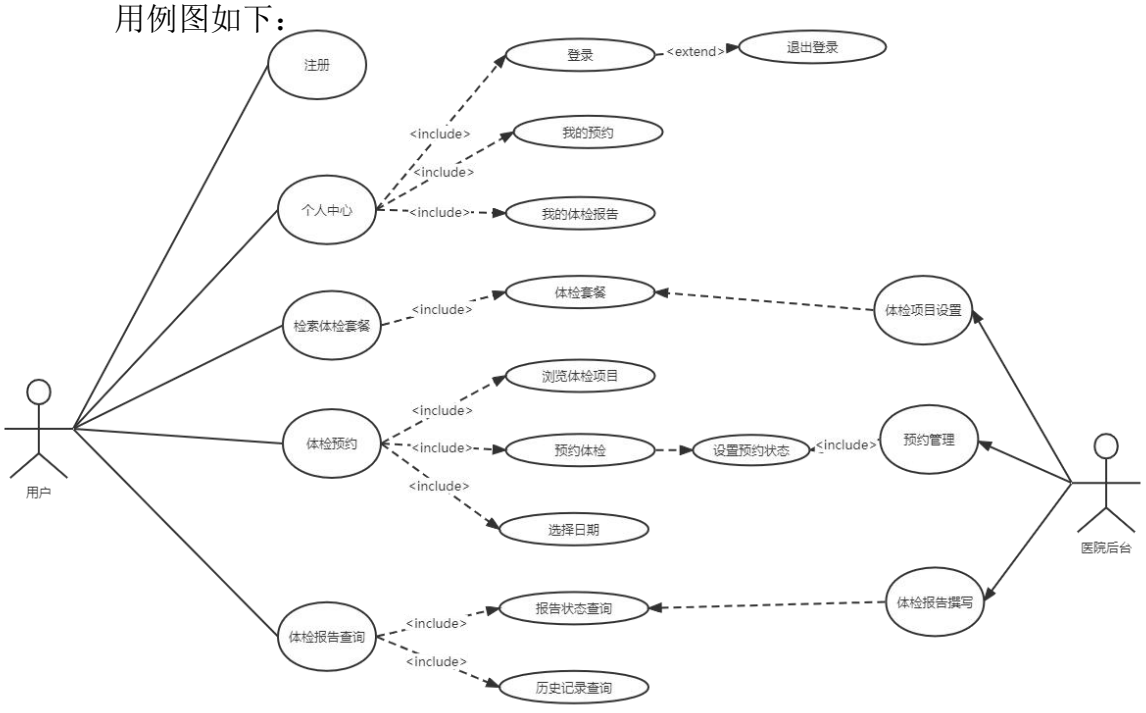
本应用致力于简化预约体检流程，方便预约操作，对线下预约体检的分流，实现体检报告的在线获取等功能。

解决实际问题

随着我国实现全面脱贫，乡村振兴战略的的不断推进，乡村的基础设施逐渐完善，尤其是普遍建成的现代化的乡村医院，乡村对于个人健康越来越重视，体检人群逐渐扩大。这个小程序结合了上述背景，考虑到疫情当下的实际情况需要减少人群在线下排队预约的情况，综合了小程序面向的对象对于智能手机了解程度的不同，开发了这款简单易用的便面体检预约小程序，主打简单，易上手，功能齐备，包含前端与后端，方便广大乡村医院与人民的体检预约需求。

需求与用例说明

用例图如下：



便民体检预约小程序的需求包含几个模块：体检预约、我的、查看体检报告、体检项目设置、预约管理和报告撰写等功能。

检索体检项目：

用户在注册登录之后，点开小程序可以在首页看到相应的医院的信息，包括医院标志，医院名称与医院的联系方式与地址，下滑就可以看到所有的体检项目。

体检预约：

用户在首页进行操作，下滑可以看到所有的体检项目，选择需要的体检项目，点击可以查看该项目的所有子项目，即该套餐下包含你的体检项目，选择预约时间后就可以点击预约。在个人中心可以查看已经进行的预约与历史预约记录。

查看体检报告：

体检结束后，在个人中心可以看到我得体检报告选项，点击体检报告选项可以看到已经完成的体检与体检报告当前状态，待后台撰写报告并上传后，用户端可以看到体检报告状态从未完成变为可查看，点击就可以查看体检报告内容。

体检项目设置：

医院有后台管理项目，体检项目设置是通过维护该医院的体检套餐表与体检项目表来实现，后台可以修改表单项来更改前端显示的体检套餐与体检项目来实现对体检项目的设置。

预约管理：

后台管理项目可以实现对体检预约的管理，当一个预约发送到后端后，会显示一个待审核的预约，后台管理的预约管理对该预约进行设置，设置为预约成功或预约失败，对应的用户个人中心的我的预约就会显示预约成功或失败。

报告撰写：

后台可以实现线上报告的撰写，当用户完成体检之后，后台可以撰写用户的体检报告，完成之后上传报告，用户端在个人中心查看我的体检报告，可以显示历史体检报告和每个体检报告的状态，报告从不可查看转为可查看后，用户可以

查看体检报告。

技术实现说明

开发语言与框架：

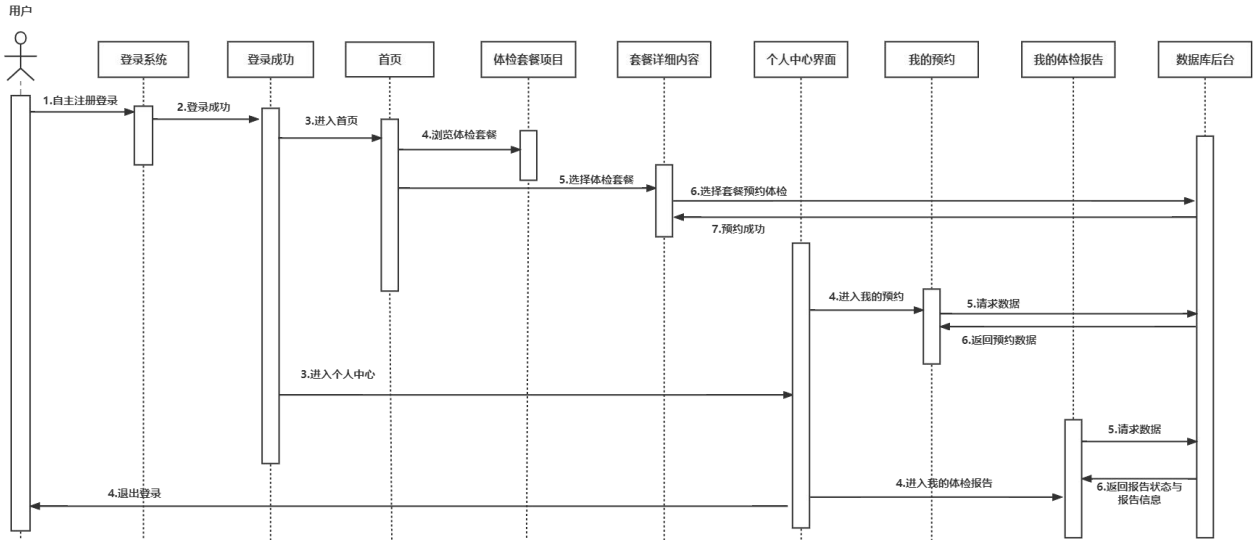
前端使用 js，vue，css 等，后端使用 springboot。

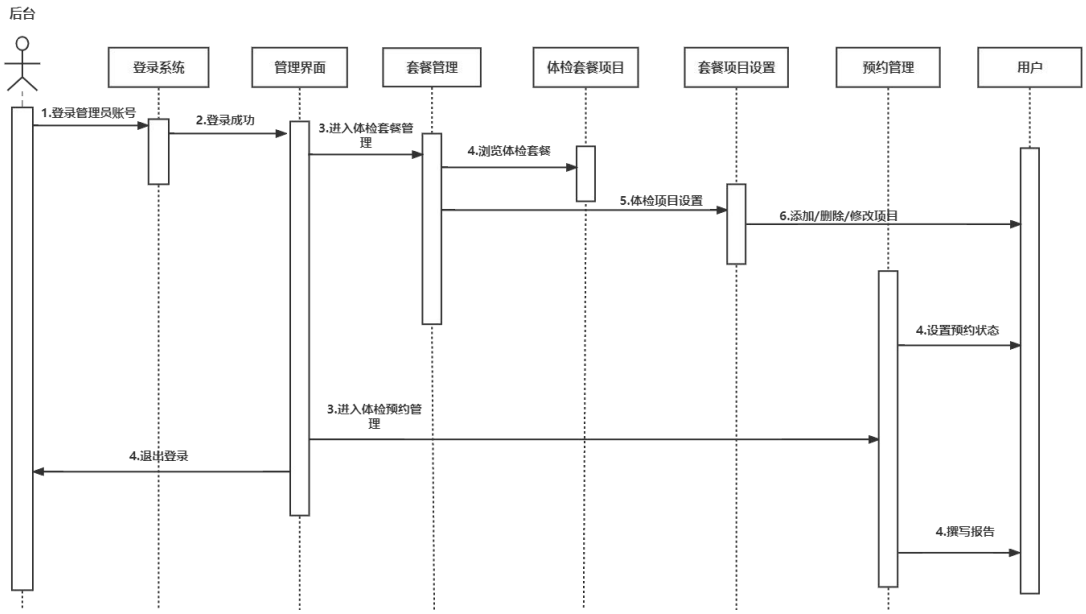
开发工具：

HBuild X，微信开发者工具

时序图

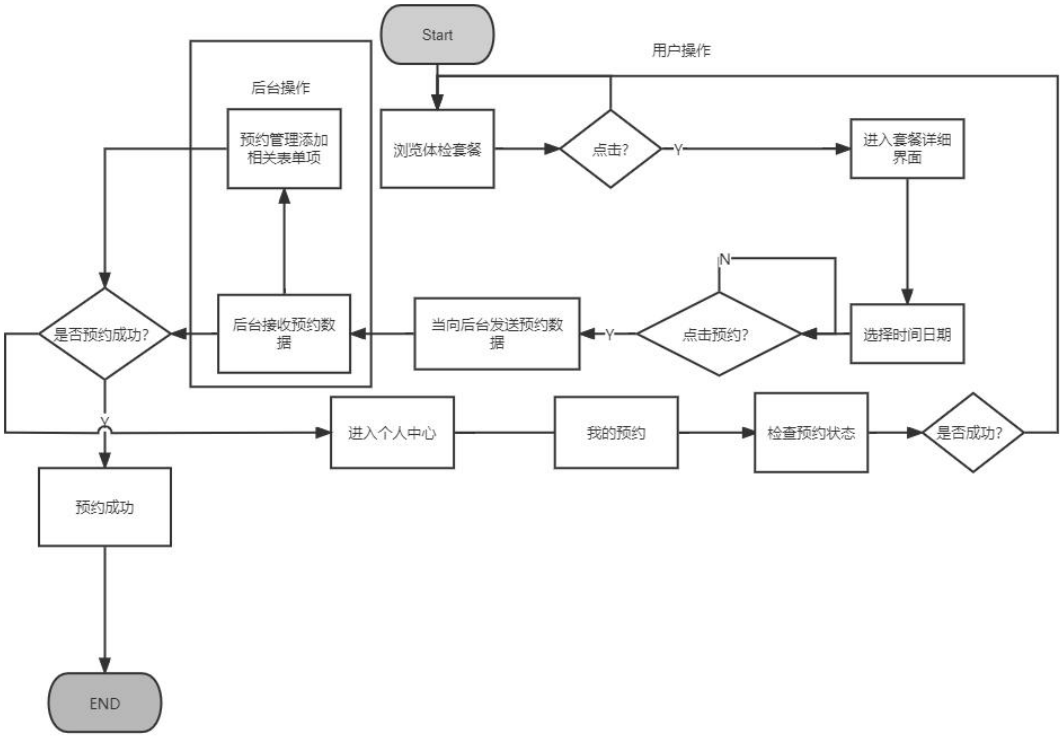
下图显示了用户与后台的操作的时序图，包含用户注册登录与体检预约、查看报告与后台的撰写报告、修改体检项目等操作。



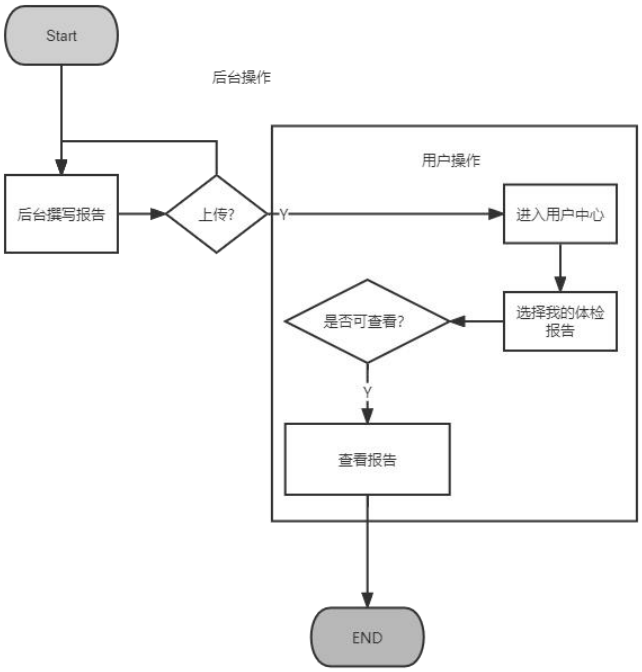


状态流转图

预约体检



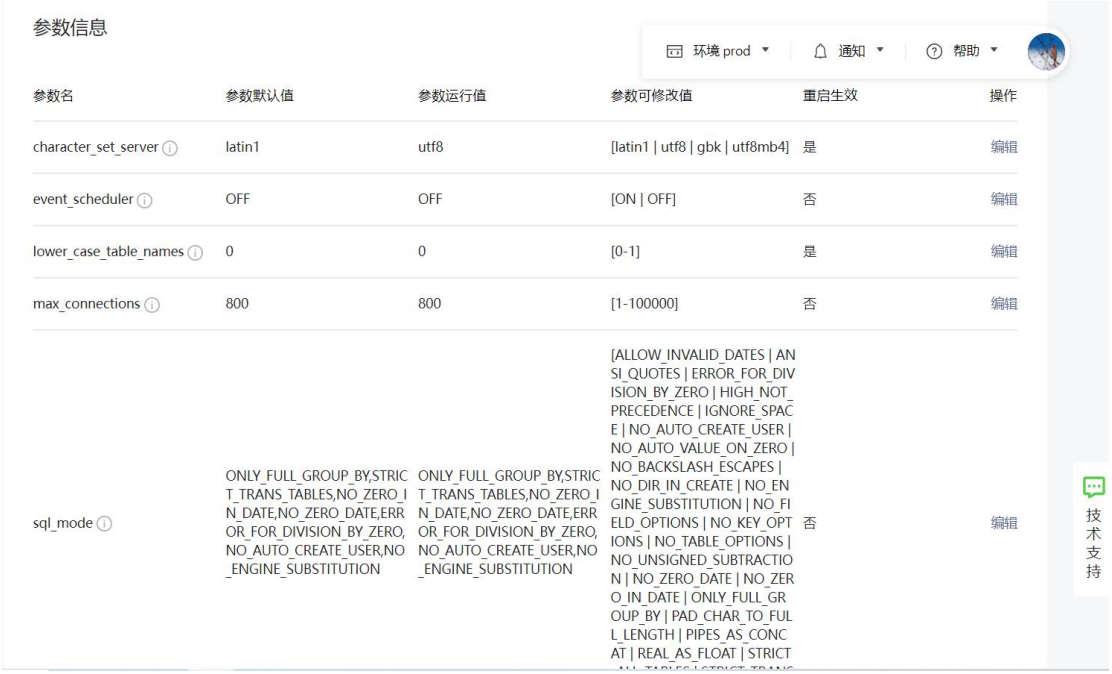
报告获取



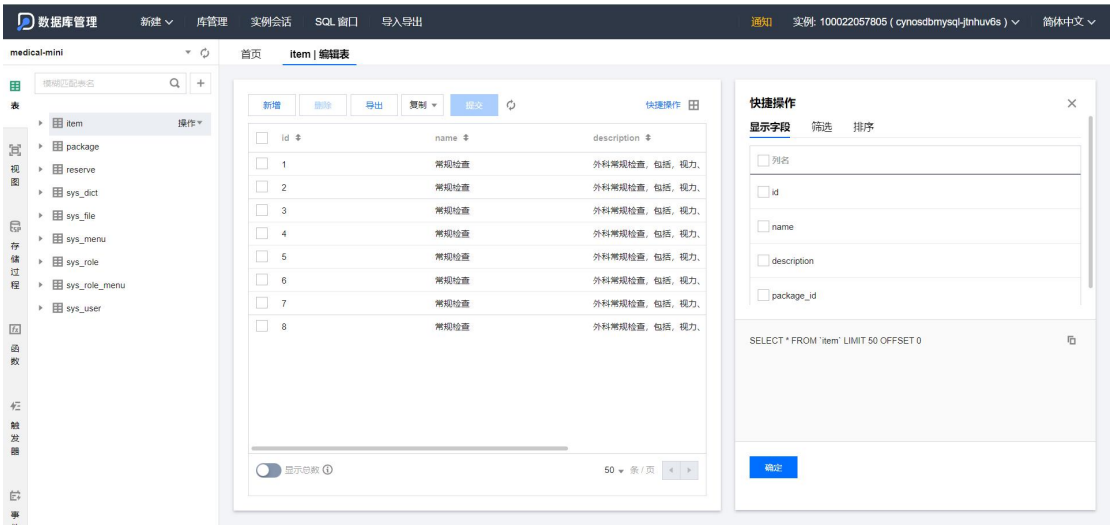
后端云托管的建立和接口调用

1、微信云托管 MySQL 数据库的建立

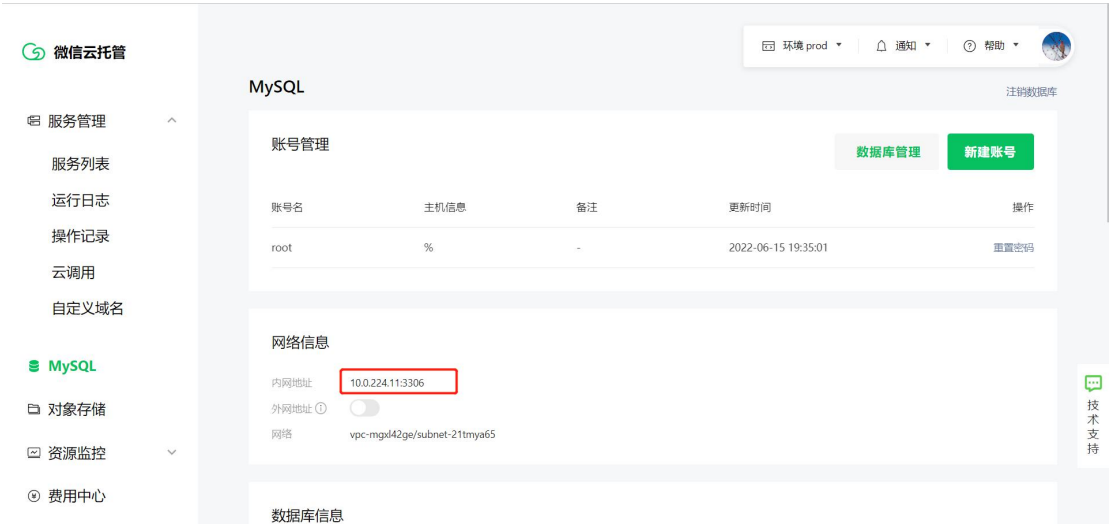
首先的一步是在微信云托管上建立数据库，以供进一步的调用。进入云托管控制台，点击左侧的 MySQL 一栏建立自己的 MySQL 数据库。数据库可以选择 MySQL8 或者 MySQL5，这里选择的是 MySQL5。可以在下拉界面看到建立的数据库信息。



建立完成后，在右上角的管理数据库中查看和管理数据库信息。根据自己的需要建立自己的数据库和数据库表。



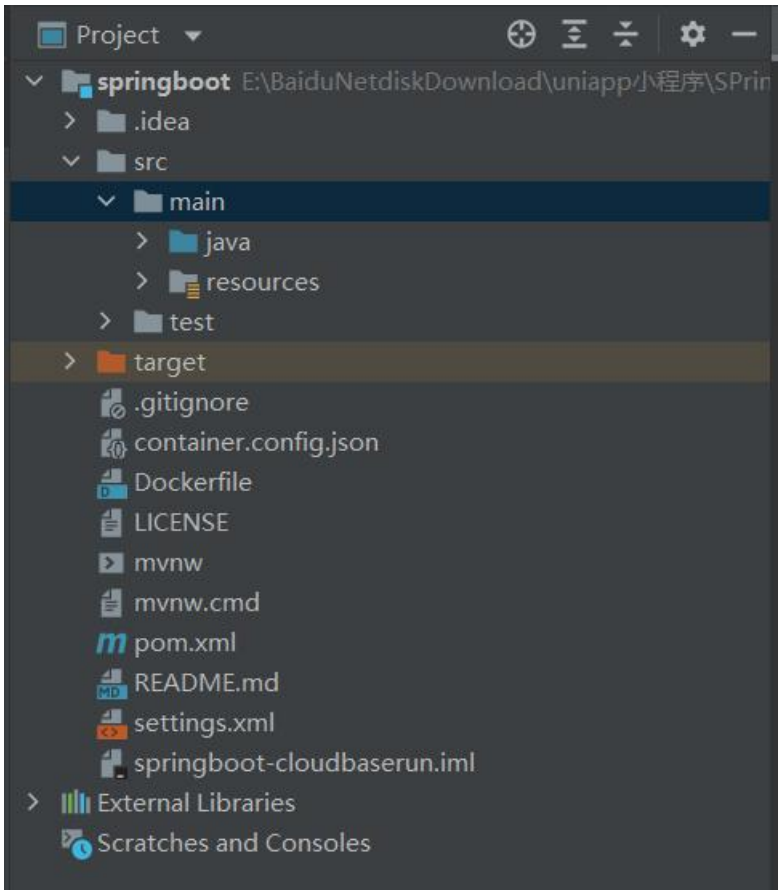
我们稍后可以请求这些信息。



这里的内网地址就是我们后面连接数据库的时候使用的地址，地址为10.0.224.11，端口号3306。

2、后端结构和设置

后端使用springboot来实现。在进行云托管前，首先进行本地后端搭建。后端的目录结构如图。



这套结构使用的是微信文档的官网模板中的springboot 框架模版：

https://github.com/WeixinCloud/wxcloudrun-springboot

我们在这个基础上进行后端的编写。

打开 application.yml 文件，可以看到：

```yaml
server:
  port: 80

spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://${MYSQL_ADDRESS}/${MYSQL_DATABASE:springboot_demo}
    username: ${MYSQL_USERNAME}
    password: ${MySQL_PASSWORD}

  mvc:
    view:
      suffix: .html
      prefix: /

mybatis:
  mapper-locations: classpath*:mapper/*Mapper.xml

```

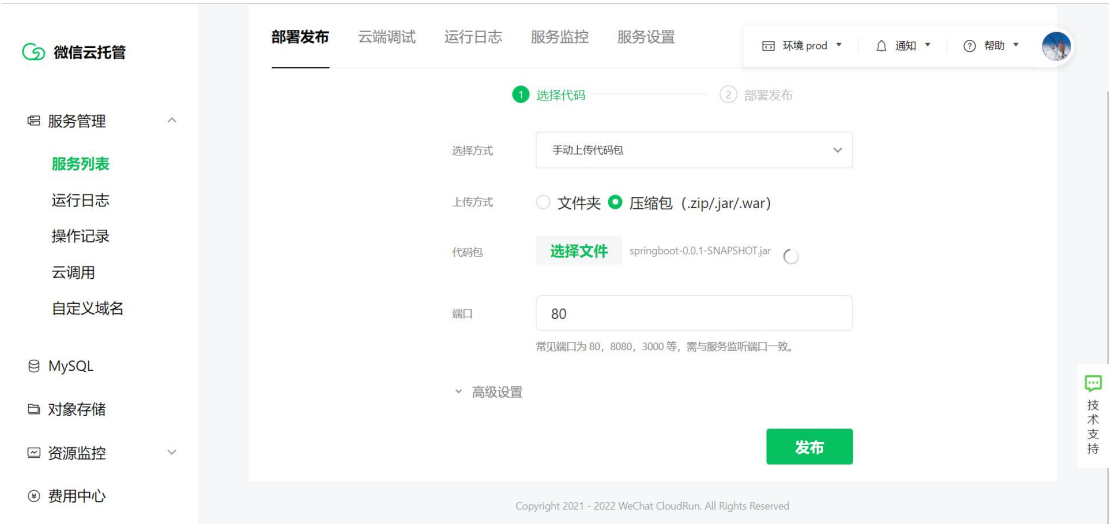
其中 MYSQL_ADDRESS 中填入上面的内网信息，也就是 10.0.224.11:3306，后面的 MYSQL_DATABASE 填入要连接的数据库名。账号和密码填写创建数据库的时候的账号和密码。当填写完成后，可以顺利连接我们刚才创建的数据库。

这里端口号设置为 80，默认即可，要与后面服务器端口号一致，否则部署的时候会失败。

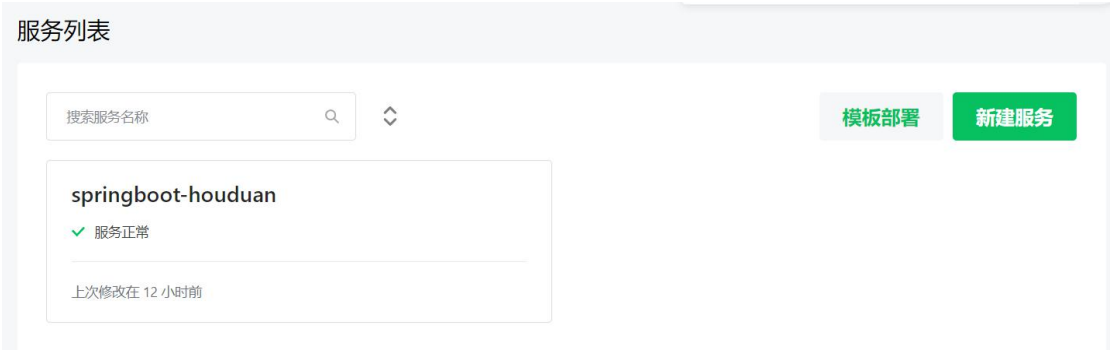
当编写好后端后，使用 mvn clean package 命令将已经写好的 springboot 打包成 jar 文件，maven 的设置文件采用上面目录结构中的 setting.xml。

3、云托管服务部署

在微信云托管-服务管理-服务列表中新建服务器，选择本地上传文件，上传刚才打包好的包进行部署。

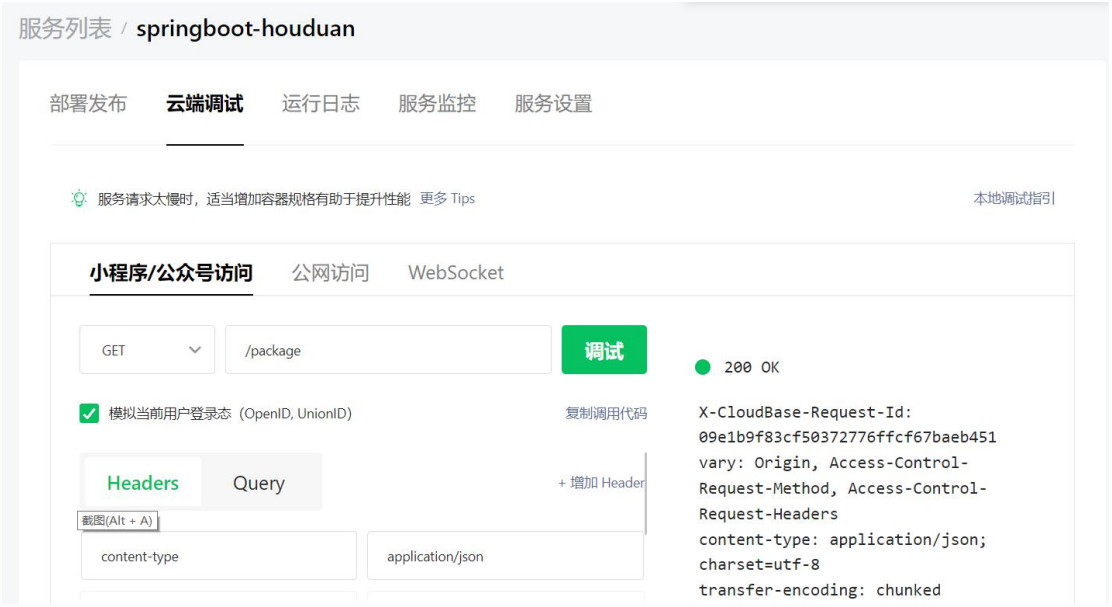


这里注意端口号，要跟后端设置的一致。点击发布，静等一会儿部署成功。



4、接口的调试和使用

首先测试部署的服务器是否可用，在服务器的云端调试里测试接口是否可用：

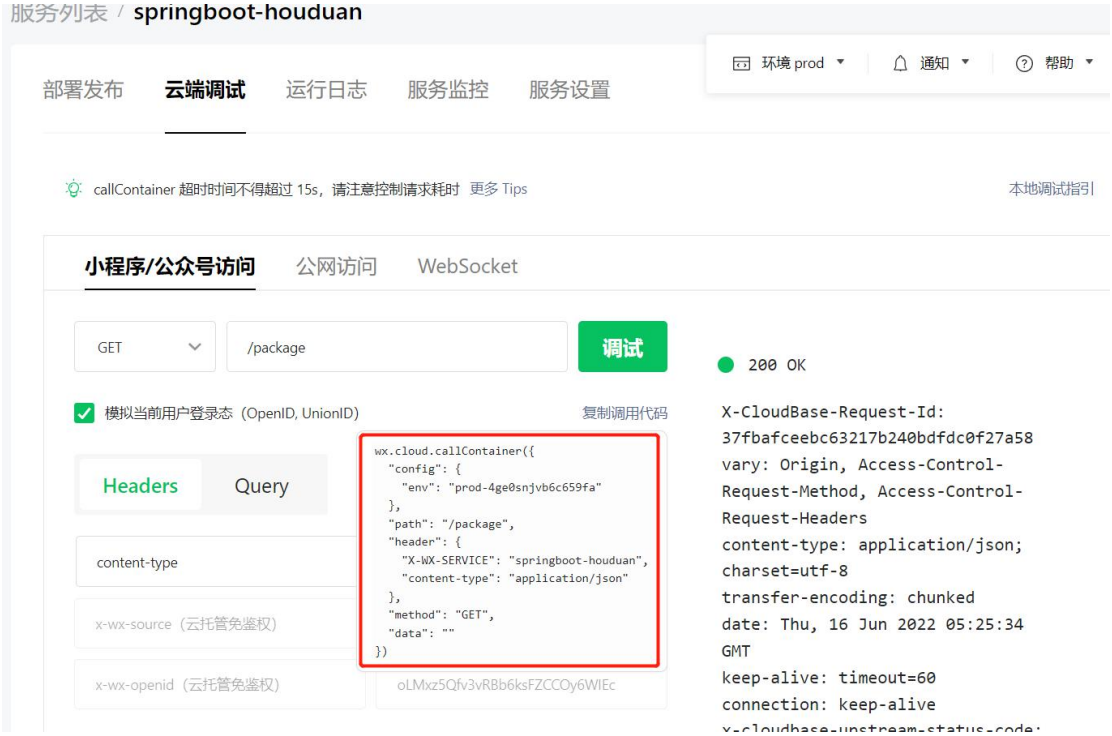


这里可以看到接口可用。然后在小程序中使用这个接口。首先要在 app.js 中做相应的配置，在 uniapp 中就是在 App.vue 中添加代码：

```vue
<script>
    export default {
        async onLaunch(){
            wx.cloud.init()
            const result = await wx.cloud.callContainer({
              config: {
                env: 'prod-4ge0snjvb6c659fa', // 微信云托管的环境ID
              },
              path: '/', // 填入业务自定义路径和参数，根目录，就是 /
              method: 'GET', // 按照自己的业务开发，选择对应的方法
              header: {
                'X-WX-SERVICE': 'springboot-houduan', // xxx中填入服务名称（微信云托管 - 服务管理 - 服务列表 - 服务名称）
              }
              // dataType:'text', // 默认不填是以 JSON 形式解析返回结果，若不想让 SDK 自己解析，可以填text
            })
            console.log(result)
            console.log('App Launch')

        } ,
```

然后在具体页面中使用，在云调用的按钮下有一个复制调用代码，实际使用的时候把这段代码复制到小程序上就行了：



为这个调用添加回调，就能把请求到的值赋予我们本地定义的变量上：这样就可以成功把数据显示在小程序上。其他的接口也同样操作。

```
},
methods: {
    load(){
        wx.cloud.callContainer({
          "config": {
            "env": "prod-4ge0snjvb6c659fa"
          },
          "path": "/package",
          "header": {
            "X-WX-SERVICE": "springboot-houduan",
            "content-type": "application/json"
          },
          "method": "GET",
          "data": ""
        }).then(res => {
            // console.log(res.data)
            this.data = res.data.data
            console.log(this.data)
        })
    }

}
```

各次迭代所实现的功能

第一次迭代：

确定选题，完成小程序的创建，完成小程序的主页面，完成后端框架基本的搭建

第二次迭代：

完成小程序各体检项目的页面，完成预约页面，完成查看结果页面，完成数据库的设计和创建。

第三次迭代：

完成大部分接口，完成登录和注册界面，完善后端，让小程序功能可以基本实现。

第四次迭代：

完成所有接口，完成后端的书写，进一步完善页面，主要是页面样式和格式，将后端云托管到服务器上，测试小程序与云托管服务器是否连接，是否可用。把最终版本上线。

后续迭代：

主要将会集中精力在完善现有功能上，也会完善页面，比如应有的配图。

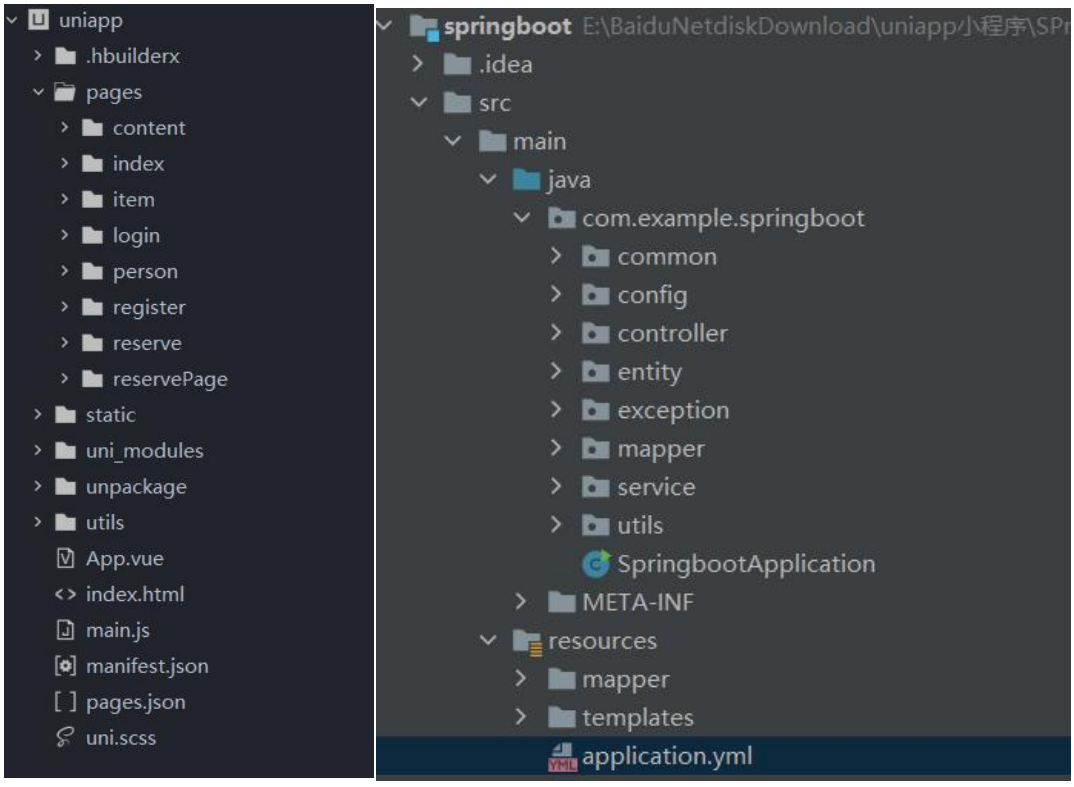
项目成果展示

优先说明：云托管服务器很不稳定，有时候需要启动服务，有时候请求会返回500，所以体验的时候有时候资源加载不出来很正常，等一会儿应该就好了（而且往往要等较长一段时间）。

AppID：wxb6456b955d56b700

体验版二维码：

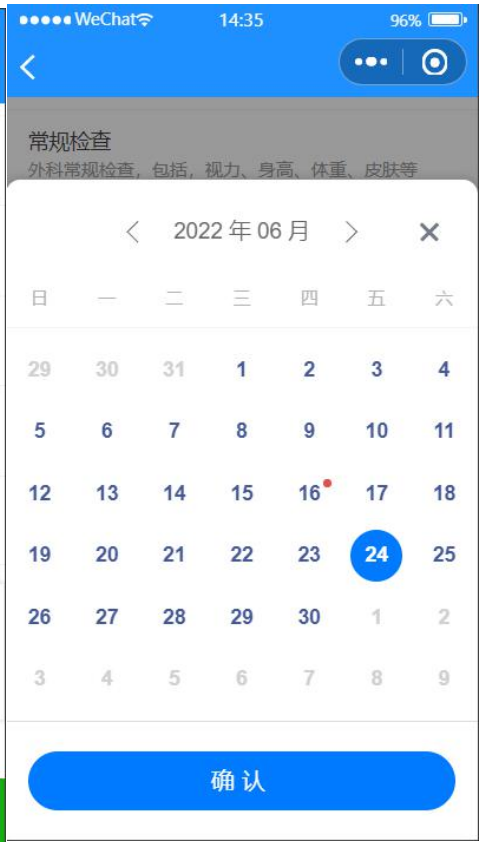
uniapp 目录结构和 springboot 目录结构：



小程序首页和个人中心页：

登录和注册页：

预约页和选择预约日期，预约日期必须在今天以后：



预约成功页和查看我的预约页：

体检报告的查看：

至此，体检预约小程序的基本功能完成。