2019-6-18

JAVA EE第四次作业

|  |  |
| --- | --- |
| 得分点 | 状态 |
| Webflux | 完成 |
| Functional handler | 完成 |
| Reactive persist | 完成 |
| Security | 完成 |

# 基于Spring Webflux的Restful API

## 获取当前所有账户

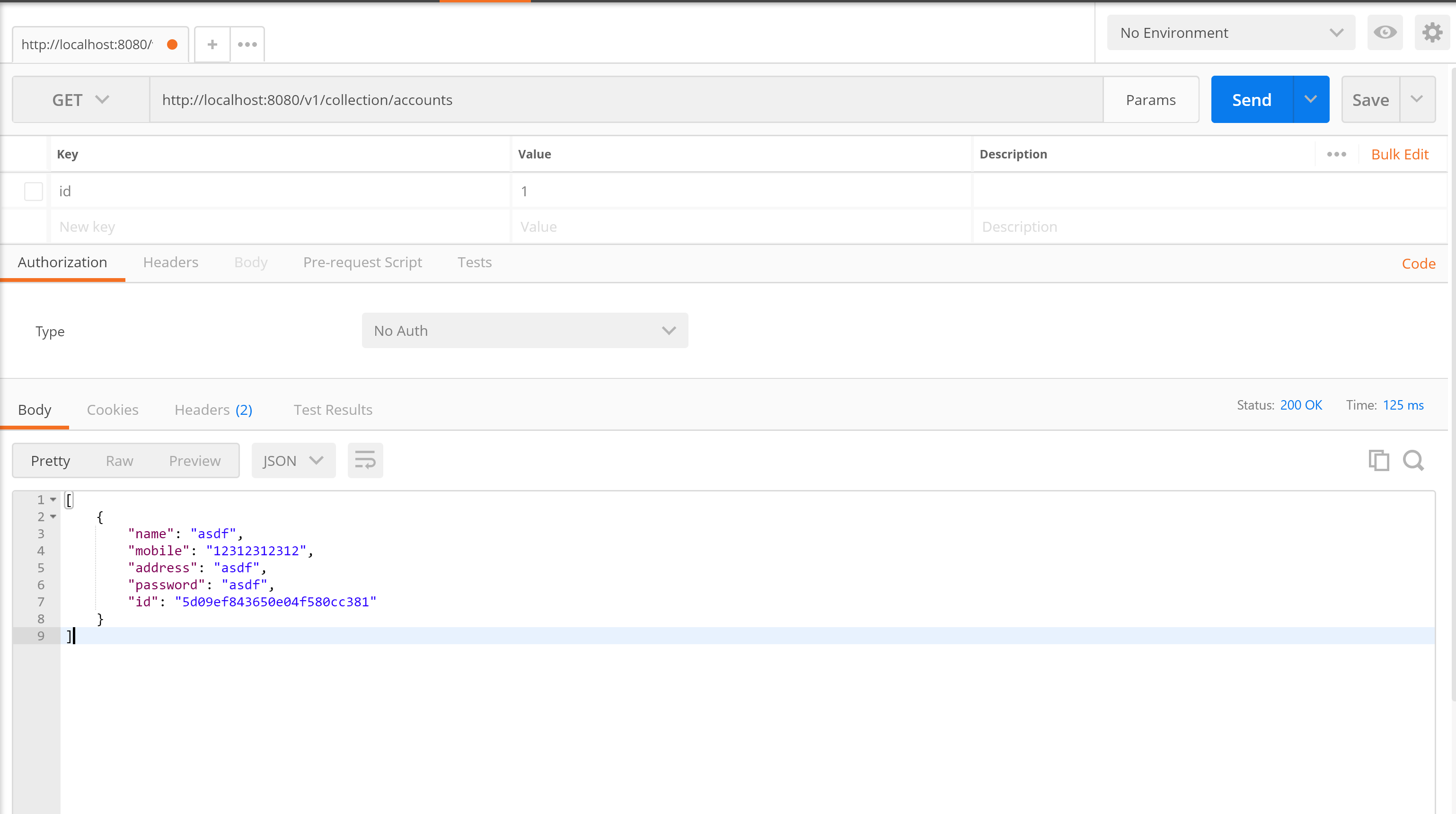
**HTTP METHOD:**

GET

**URI:**

/v1/collection/accounts

**Demo:**



## 增加一个账户

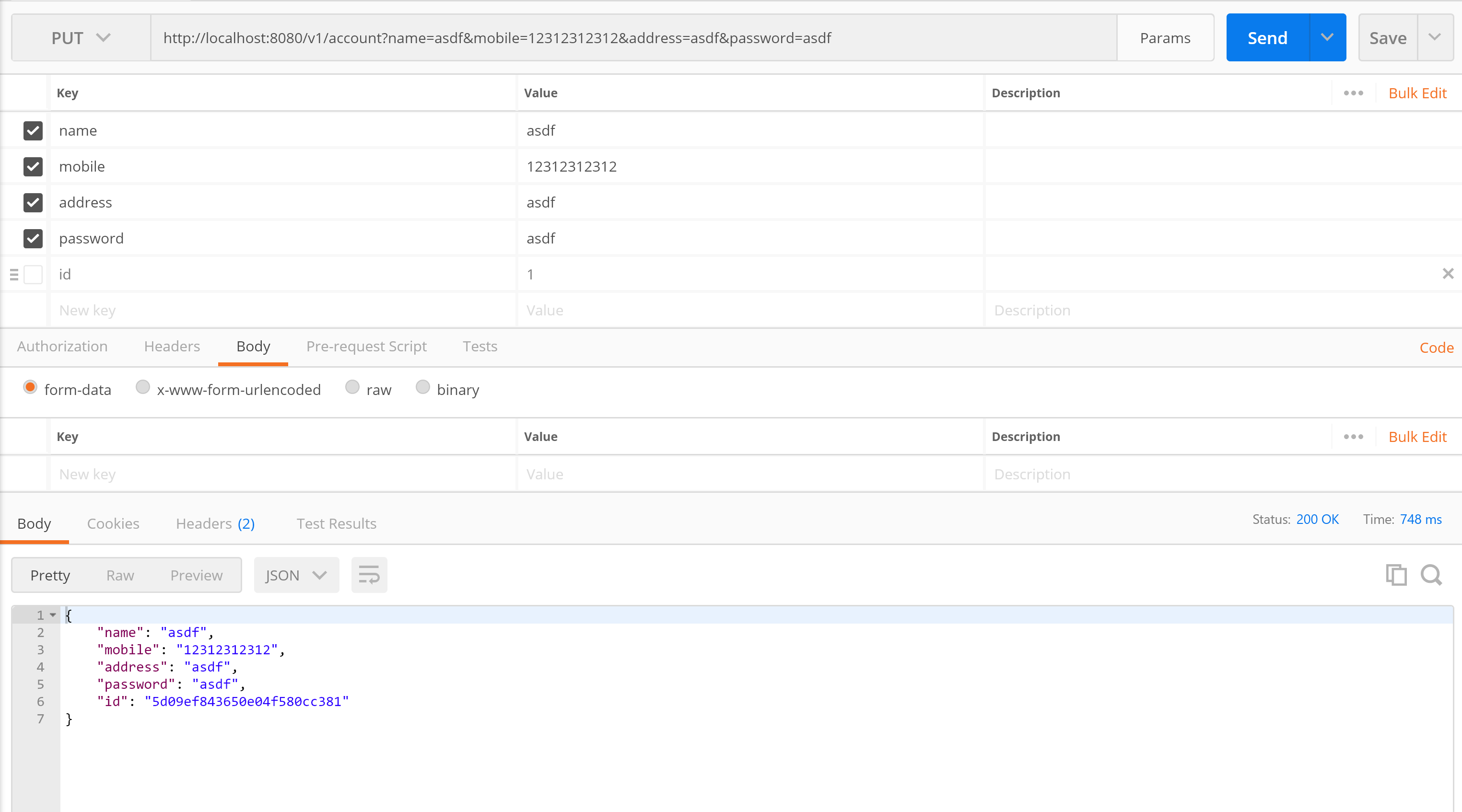
**HTTP METHOD:**

PUT

**URI:**

/v1/account

**Demo:**



## 根据用户名删除一个账户

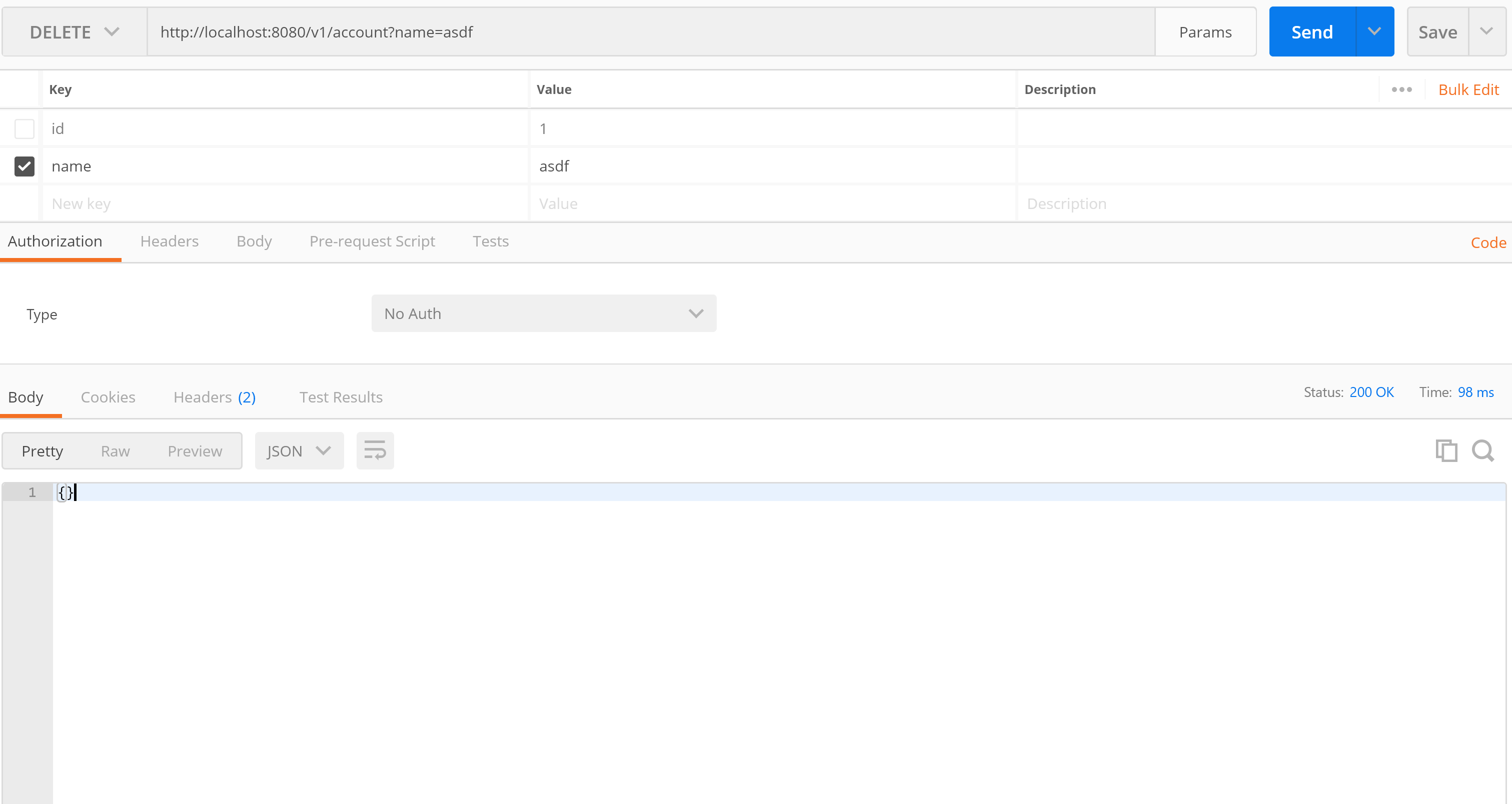
**HTTP METHOD:**

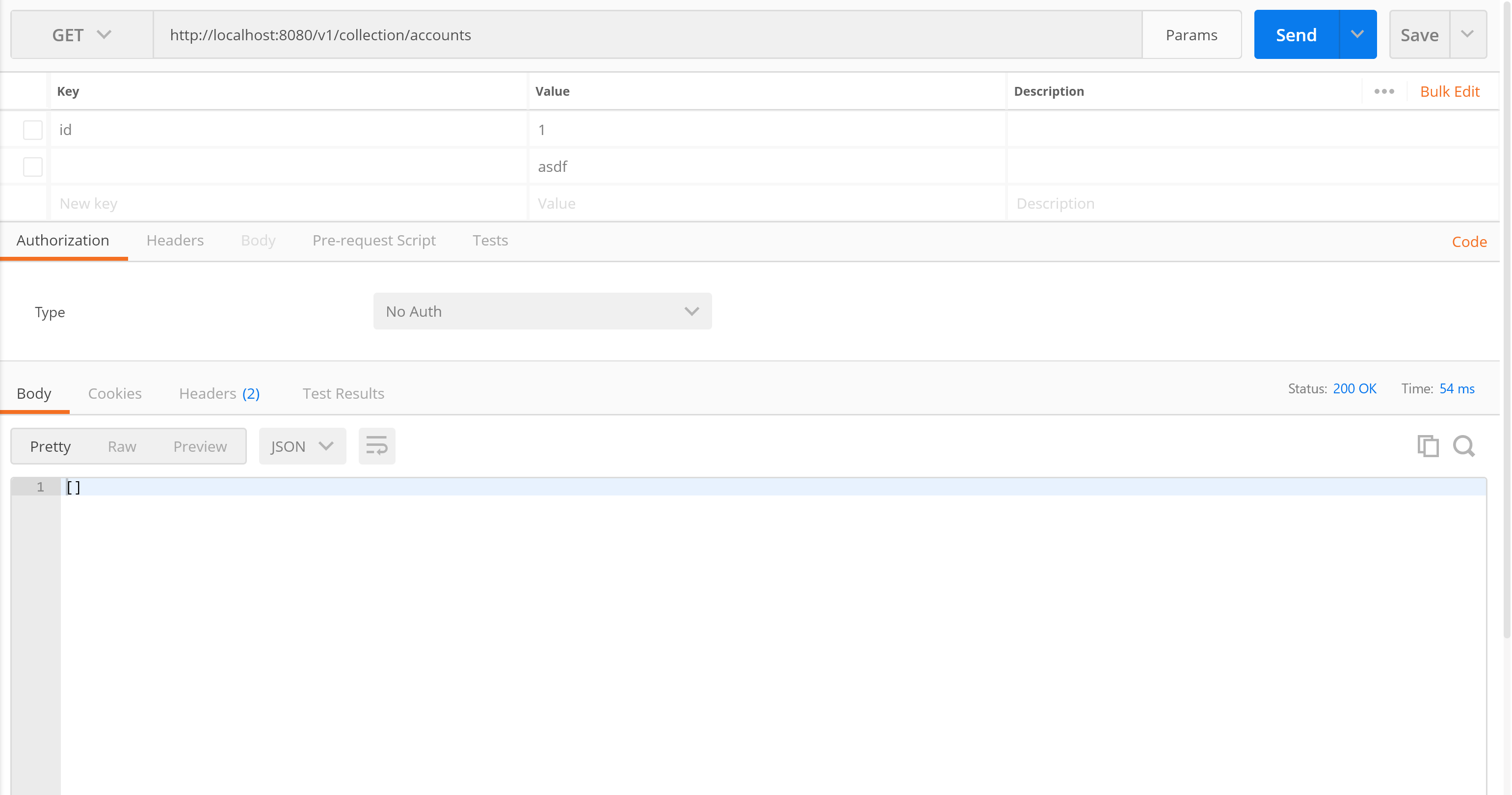
DELETE

**URI:**

/v1/account

**Demo:**





## 根据用户名寻找一个账户

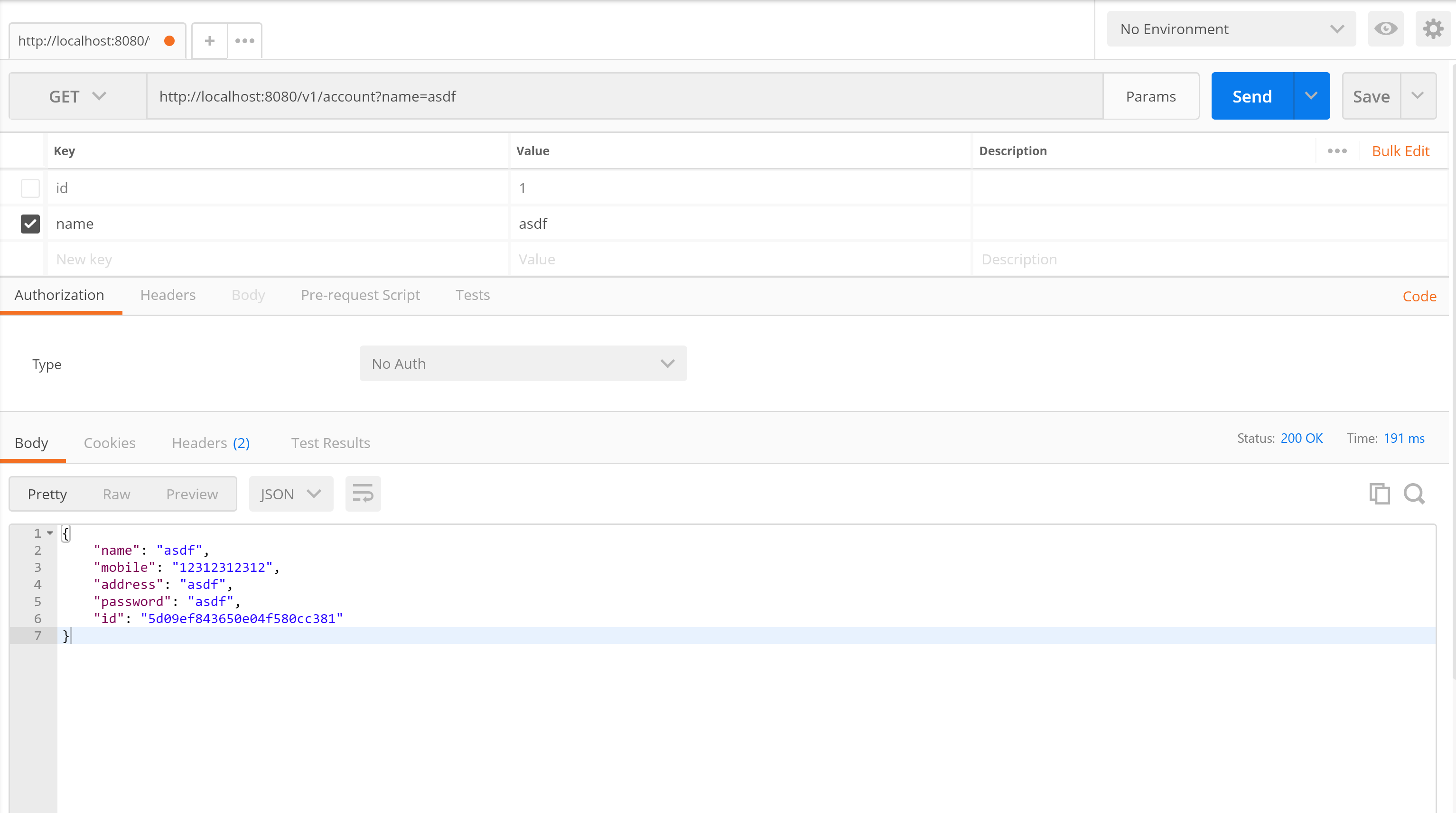
**HTTP METHOD:**

GET

**URI:**

/v1/account

**Demo:**



## 获取当前所有教练

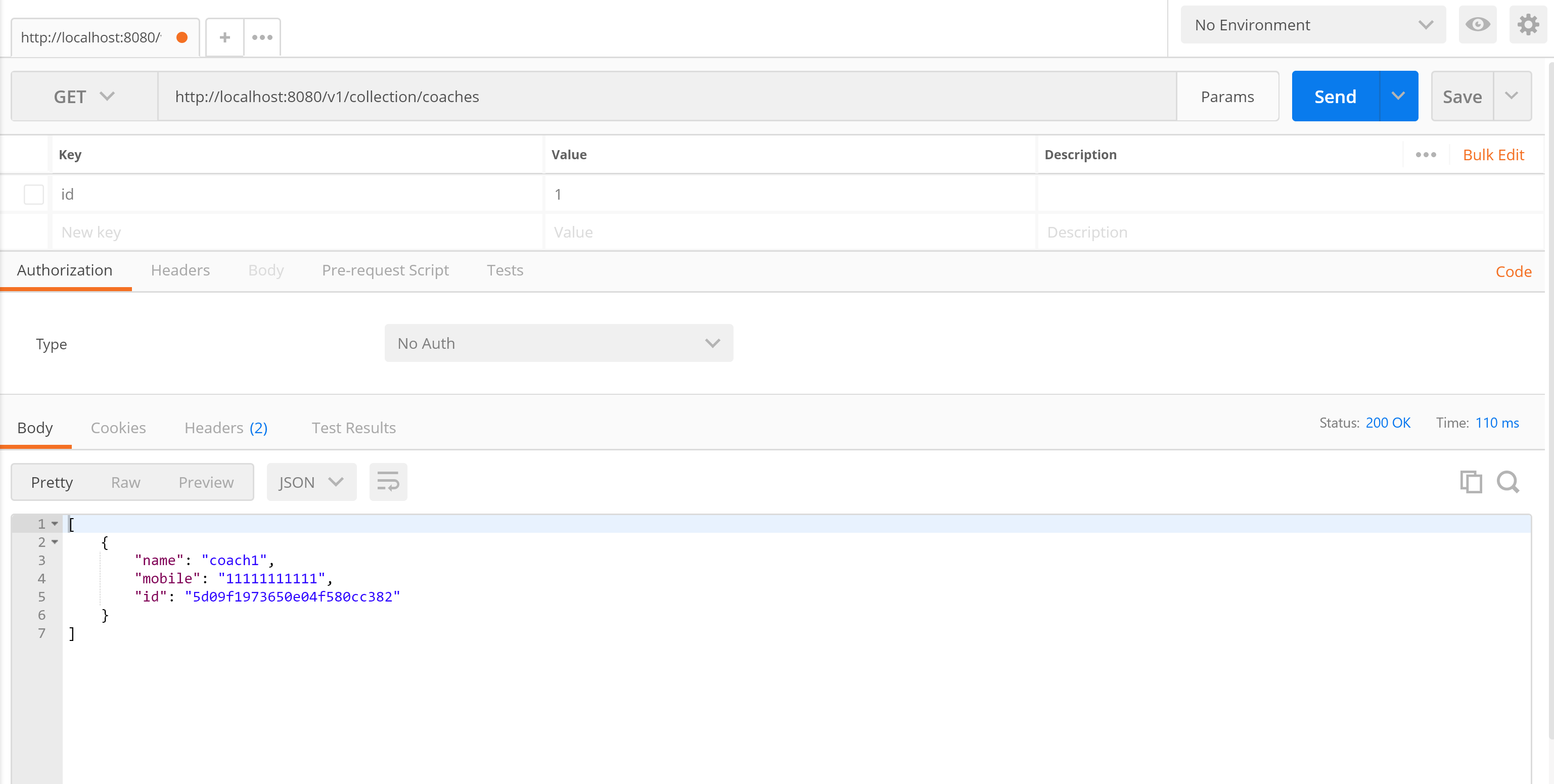
**HTTP METHOD:**

GET

**URI:**

/v1/collection/coaches

**Demo:**



## 增加一个教练

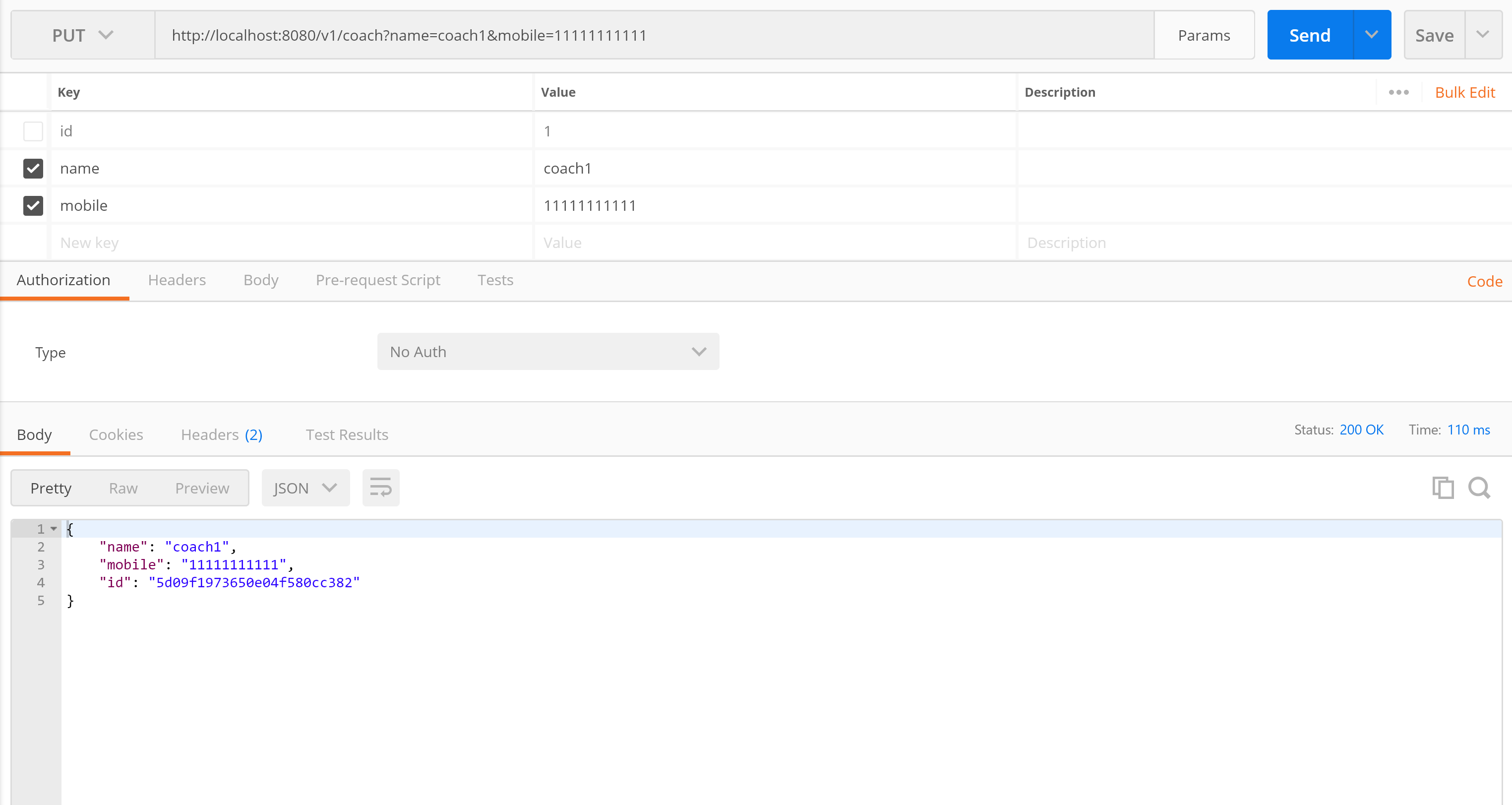
**HTTP METHOD:**

PUT

**URI:**

/v1/coach

**Demo:**



## 根据名字删除一个教练

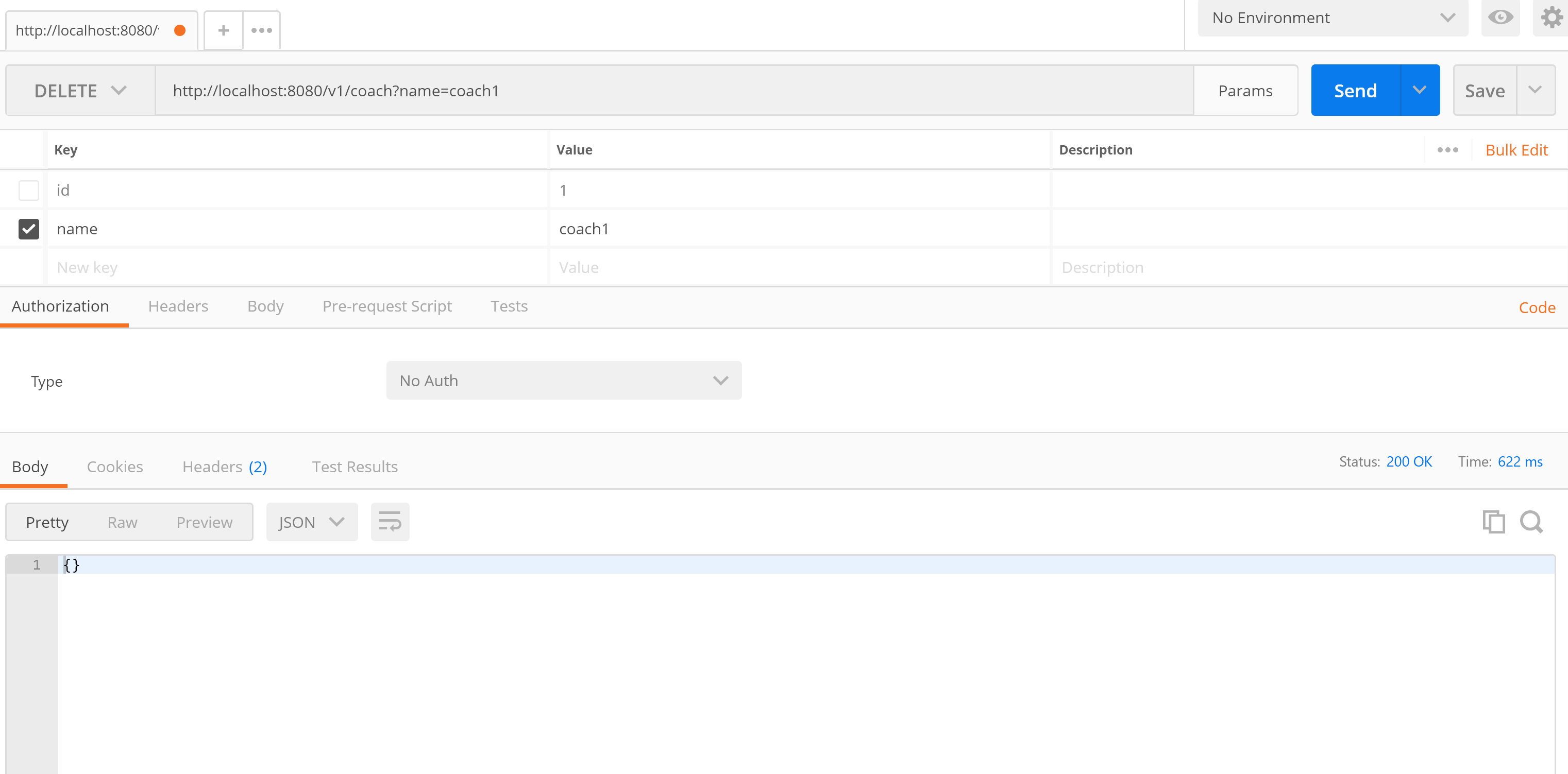
**HTTP METHOD:**

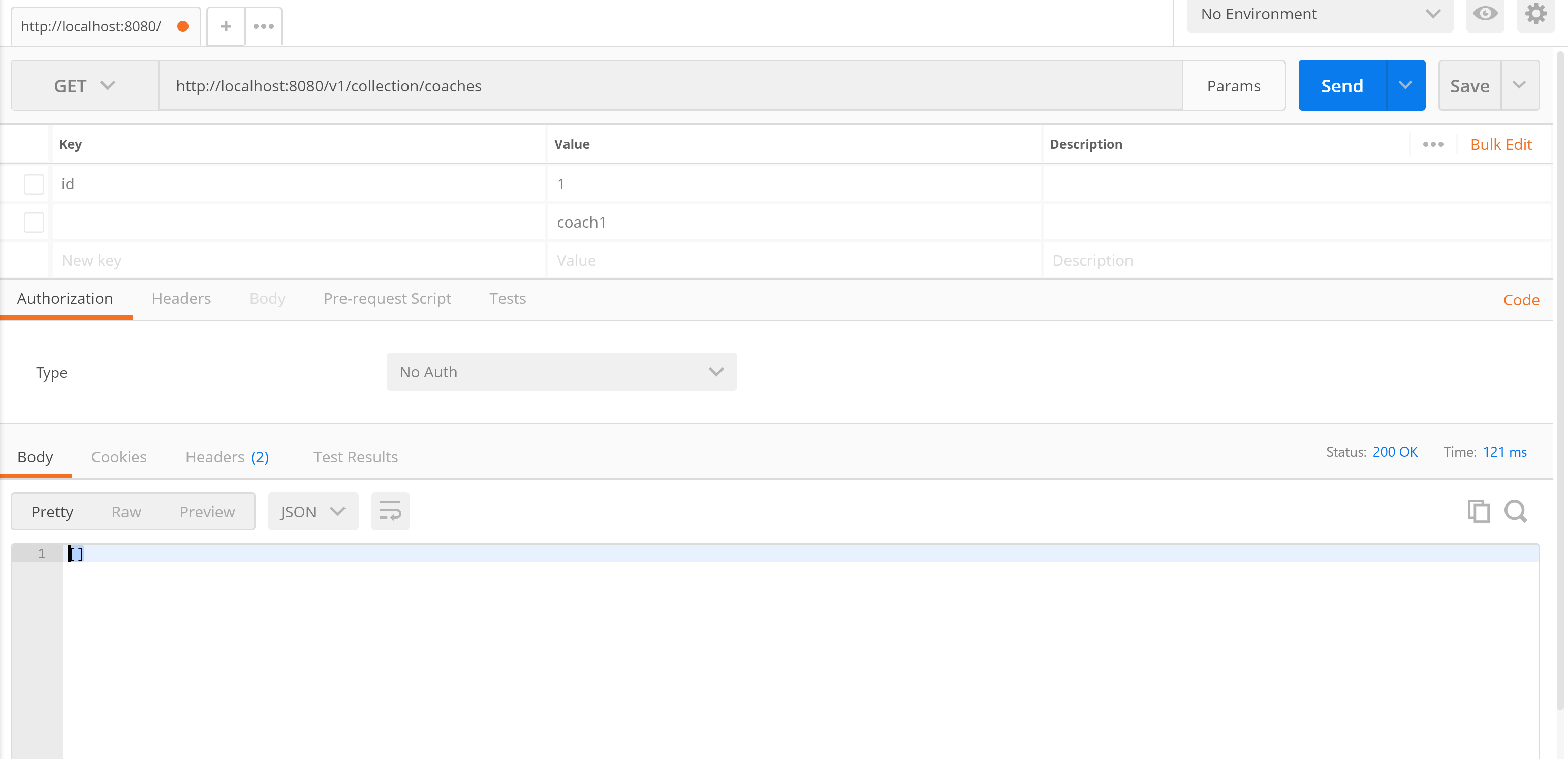
DELETE

**URI:**

/v1/coach

**Demo:**





## 根据名字寻找一个教练

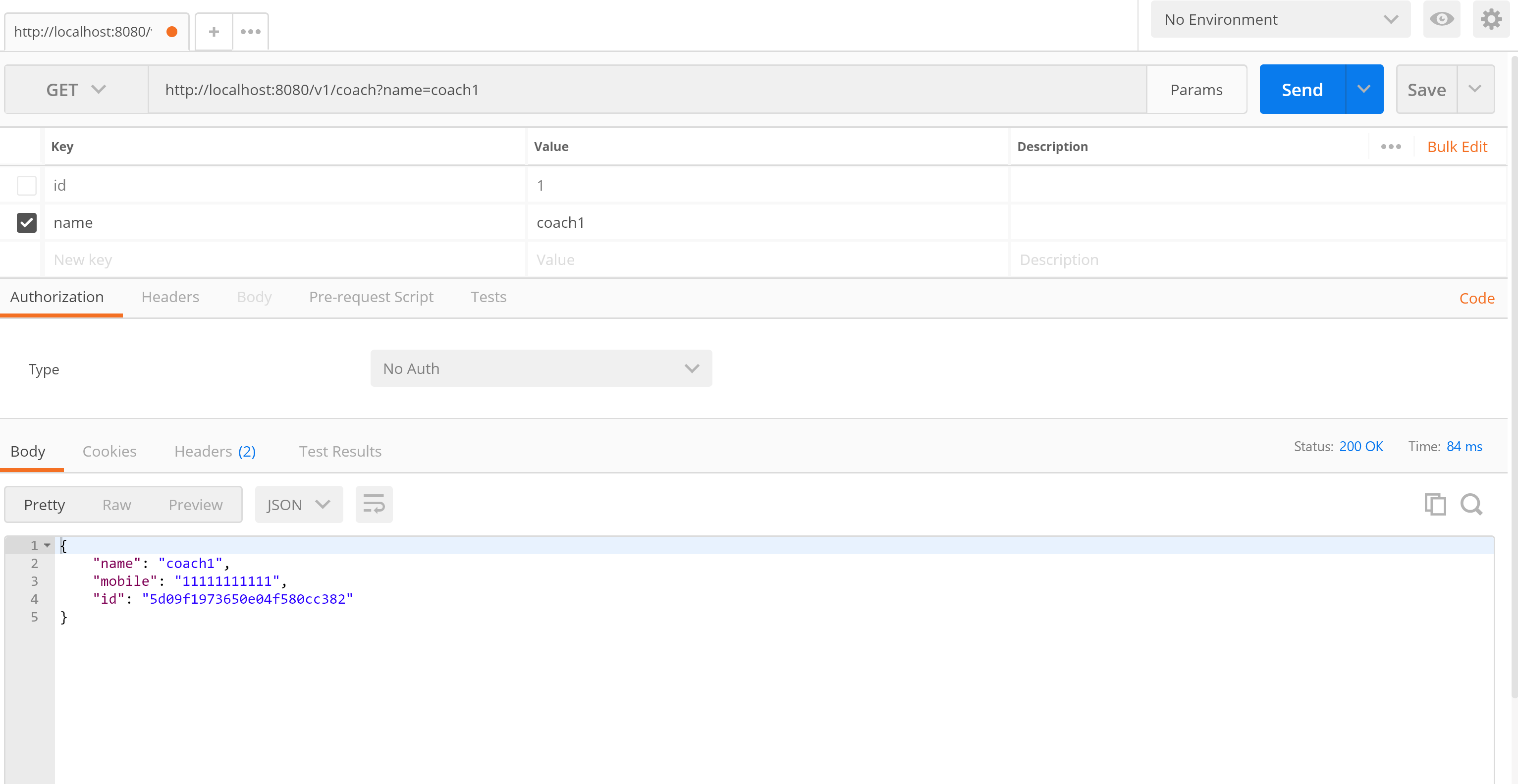
**HTTP METHOD:**

GET

**URI:**

/v1/coach

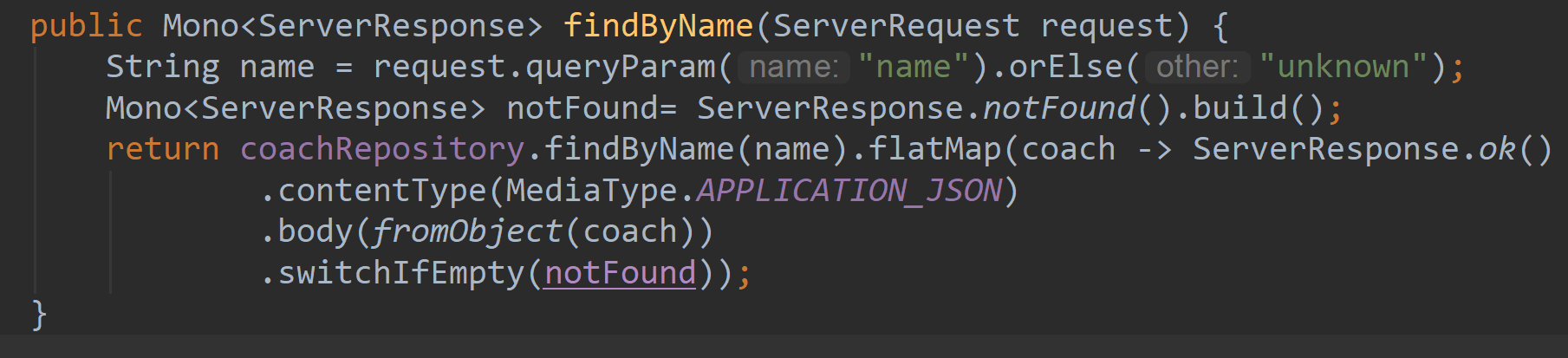
**Demo:**



# 基于Handler与Route的函数式编程

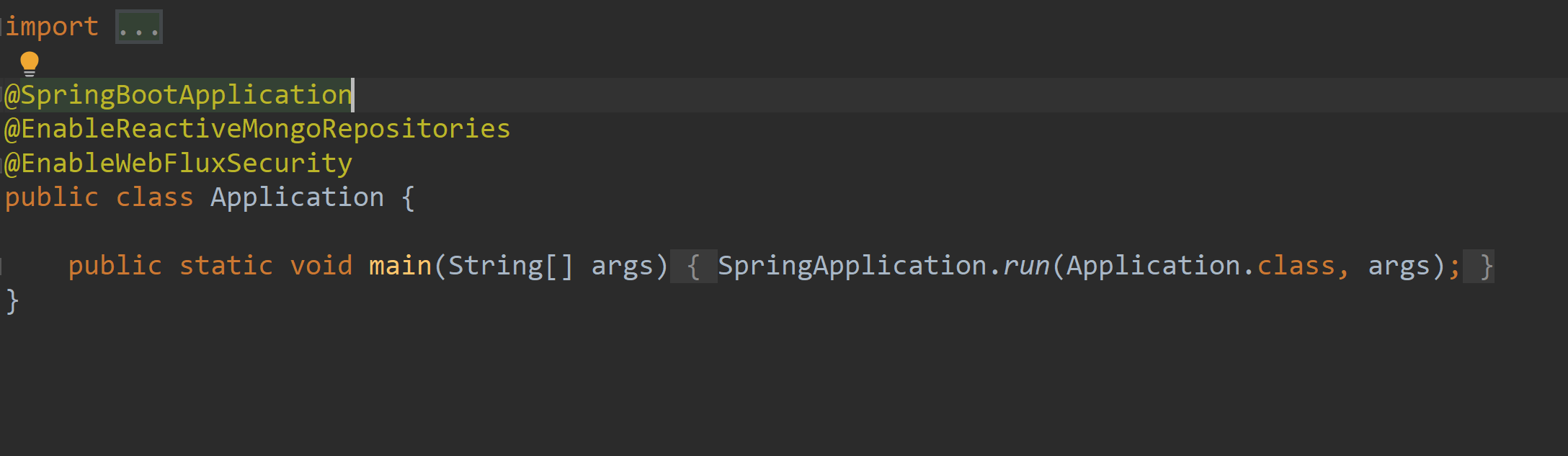
本项目主要使用了Spring Webflux，并利用了函数式编程技巧。在Routes中，将特定的Route与对应的Handler完成绑定，处理相应的请求。

如图所示，为CoachHandler中的findByName方法，该方法负责处理HTTP Method为GET，URI为/v1/coach的请求，将会根据传过来的name参数，寻找到相应的用户信息并发布。首先，在绑定Handler与Route时，已经进行了函数式编程，其次，该图中的返回结果，也采用了函数式编程进行处理。

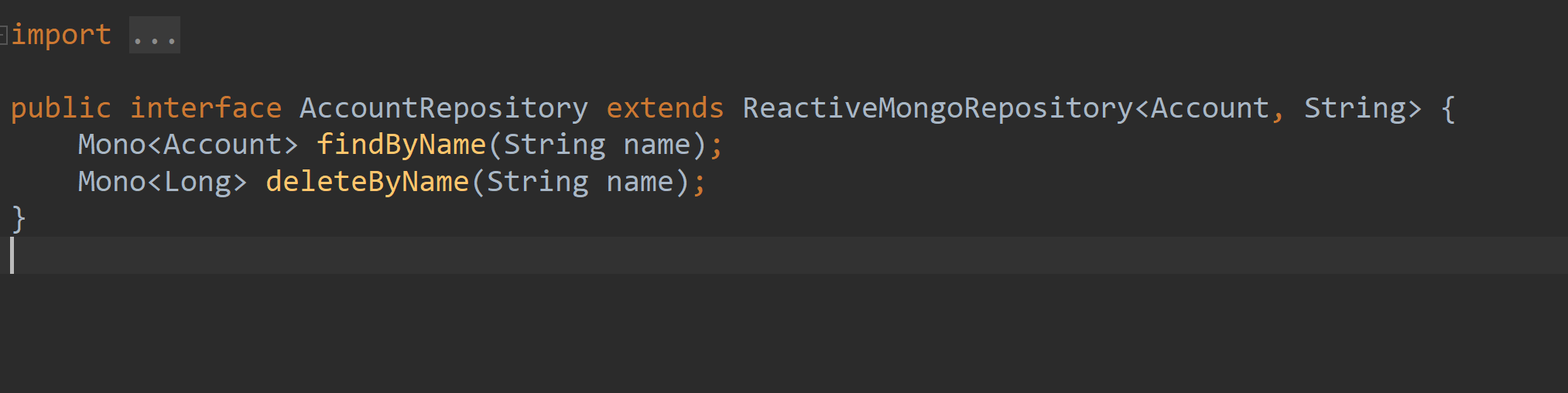


# 基于MongoDB的Reactive数据持久化

本项目使用了MongoDB，同时进行了Reactive类型的数据持久化操作，从图中可以看出，Reactive操作主要依赖于框架spring-boot-starter-data-mongodb-reactive。图中使用了EnableReactiveMongoRepositories注解，来启动框架功能。



于此同时，已有的Repositories继承了ReactiveMongoRepository，并通过该方式完成持久化层的Reactive化。



# 4 基于Spring Security的访问控制

我们使用了Spring Security框架来对项目进行访问控制，避免非法用户登入并使用关键API（如PUT、POST、DELETE等对数据造成影响的API）。

我们通过Configuration注解，应用对Security框架的配置，然后细化UserDetailsRepository，最终完成了项目的权限控制，保证了API的安全性。

