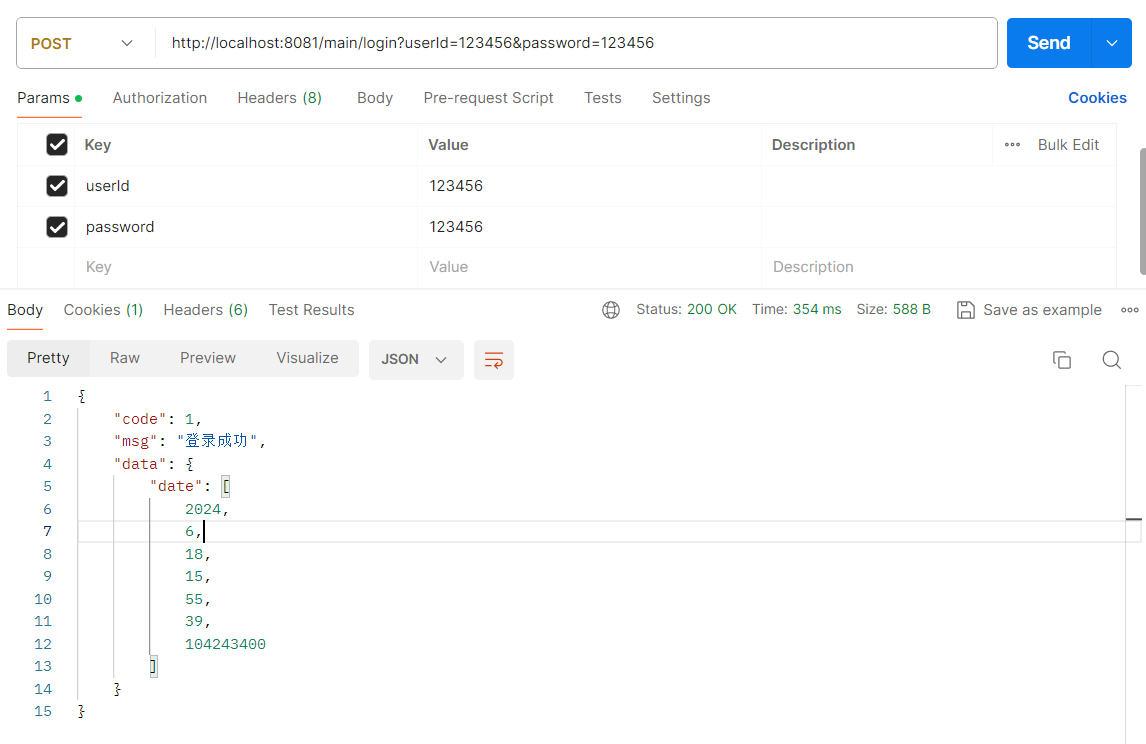
**Login接口测试**

<http://localhost:8081/main/login?userId=123456&password=123456>

userId、password用自己的数据



**Regist接口测试（user表中没有age属性的会报错，age是后面加的，相应的entity、xml映射文件需要修改，不然也会报错）**

**entity中的user表：**

package org.example.keephealthy02.Entity;

import com.baomidou.mybatisplus.annotation.TableName;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import java.time.LocalDate;

import java.util.Date;

@Data

@AllArgsConstructor

@NoArgsConstructor

@TableName("user")

public class User {

private String id;

private String name;

private String password;

private Integer sex;

private Integer age;

private LocalDate birthday;

private Double height;

private Double weight;

private Double targetweight;

private String pastMedicalHistory;

}

**resources中的Mapper文件UserMapper.xml映射文件：**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="org.example.keephealthy02.Mapper.UserMapper">

<resultMap id="BaseResultMap" type="org.example.keephealthy02.Entity.User">

<id column="id" jdbcType="VARCHAR" property="id" />

<result column="name" jdbcType="VARCHAR" property="name" />

<result column="password" jdbcType="VARCHAR" property="password" />

<result column="sex" jdbcType="INTEGER" property="sex" />

<result column="age" jdbcType="INTEGER" property="age" />

<result column="birthday" jdbcType="DATE" property="birthday" />

<result column="height" jdbcType="REAL" property="height" />

<result column="weight" jdbcType="REAL" property="weight" />

<result column="targetweight" jdbcType="VARCHAR" property="targetweight" />

</resultMap>

<resultMap extends="BaseResultMap" id="ResultMapWithBLOBs" type="org.example.keephealthy02.Entity.User">

<result column="past\_medical\_history" jdbcType="LONGVARCHAR" property="pastMedicalHistory" />

</resultMap>

<sql id="Base\_Column\_List">

id, name, password, sex, age,birthday, height, weight, targetweight

</sql>

<sql id="Blob\_Column\_List">

past\_medical\_history

</sql>

<select id="selectById" resultType="org.example.keephealthy02.Entity.User" parameterType="java.lang.String">

select \* from user

where id = #{id}

</select>

<delete id="deleteById" parameterType="java.lang.String">

delete from user

where id = #{id,jdbcType=VARCHAR}

</delete>

<insert id="insert" parameterType="org.example.keephealthy02.Entity.User">

insert into user (id, name, password,

sex, age,birthday, height, weight,

targetweight, past\_medical\_history)

values (#{id,jdbcType=VARCHAR}, #{name,jdbcType=VARCHAR}, #{password,jdbcType=VARCHAR},

#{sex,jdbcType=INTEGER},#{age,jdbcType=INTEGER}, #{birthday,jdbcType=DATE}, #{height,jdbcType=REAL}, #{weight,jdbcType=REAL},

#{targetweight,jdbcType=VARCHAR}, #{pastMedicalHistory,jdbcType=LONGVARCHAR})

</insert>

<update id="updateById" parameterType="org.example.keephealthy02.Entity.User">

update user

<set>

<if test="et.name != null">

name = #{et.name,jdbcType=VARCHAR},

</if>

<if test="et.password != null">

password = #{et.password,jdbcType=VARCHAR},

</if>

<if test="et.sex != null">

sex = #{et.sex,jdbcType=INTEGER},

</if>

<if test="et.age != null">

sex = #{et.age,jdbcType=INTEGER},

</if>

<if test="et.birthday != null">

birthday = #{et.birthday,jdbcType=DATE},

</if>

<if test="et.height != null">

height = #{et.height,jdbcType=REAL},

</if>

<if test="et.weight != null">

weight = #{et.weight,jdbcType=REAL},

</if>

<if test="et.targetweight != null">

targetweight = #{et.targetweight,jdbcType=VARCHAR},

</if>

<if test="et.pastMedicalHistory != null">

past\_medical\_history = #{et.pastMedicalHistory,jdbcType=LONGVARCHAR},

</if>

</set>

where id = #{et.id,jdbcType=VARCHAR}

</update>

</mapper>

<http://localhost:8081/main/regist>

例子：

{

    "id": "1",

    "name": "XXX",

    "password": "123456",

    "age":20,

    "birthday": "1990-01-01",

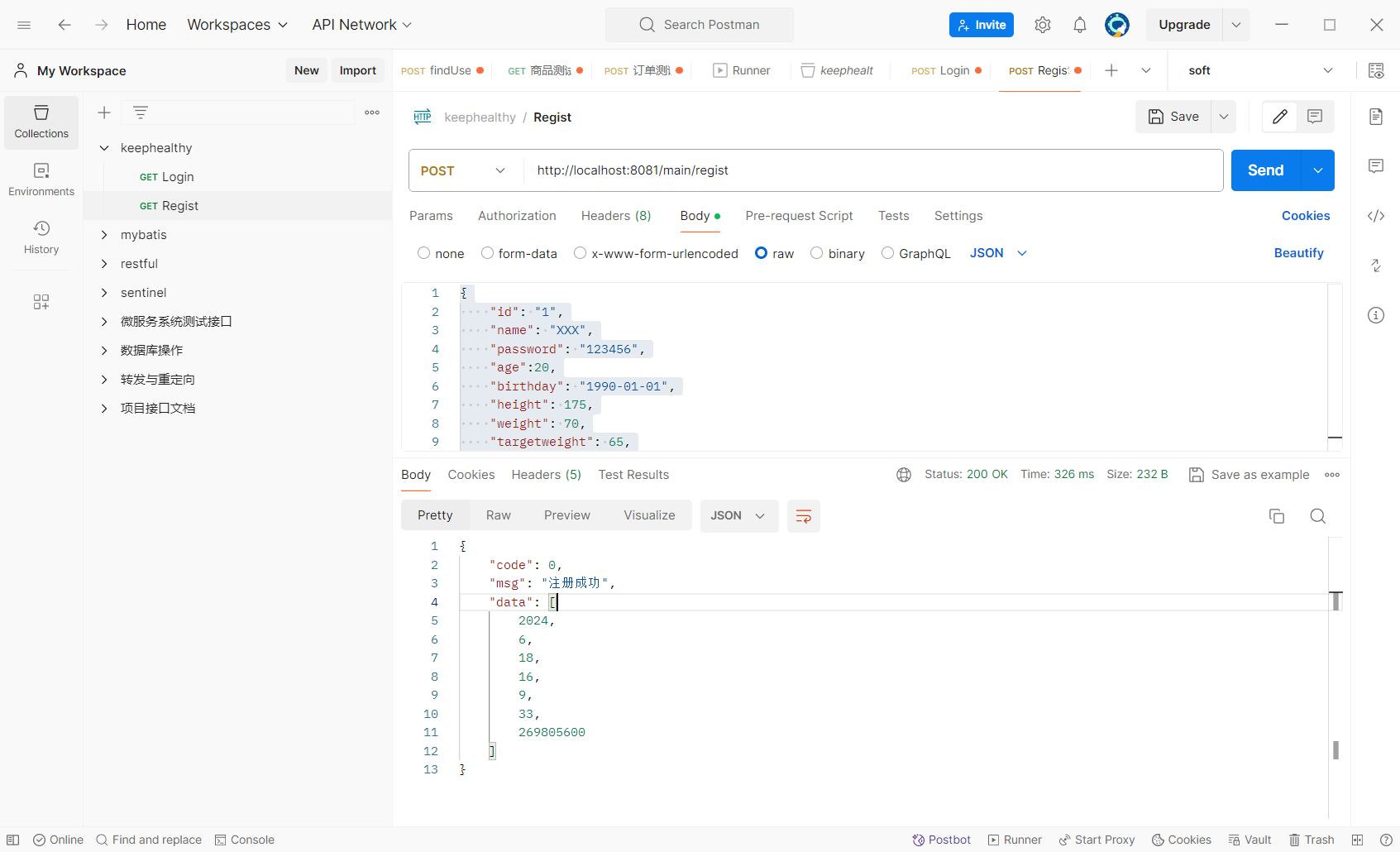
    "height": 175,

    "weight": 70,

    "targetweight": 65,

    "pastMedicalHistory": "None"

}



**Update接口测试（需要拿到Login的token参数）**

[http://localhost:8081/main/update](http://localhost:8081/main/regist)

例子：

{

    "id": "1",

    "name": "XXX",

    "password": "123456",

    "age":20,

    "birthday": "2004-01-01",

    "height": 175,

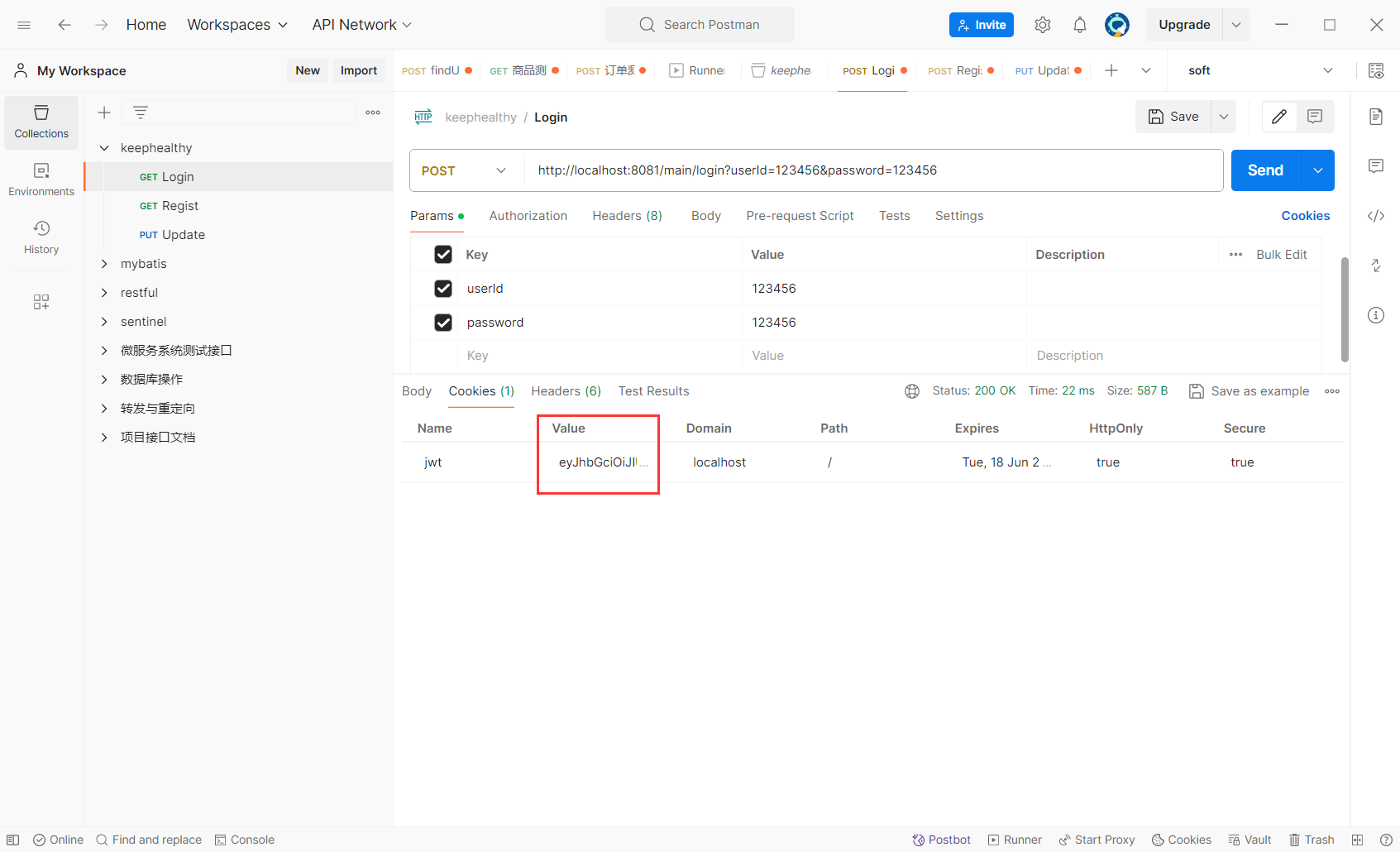
    "weight": 70,

    "targetweight": 65,

    "pastMedicalHistory": "None"

}

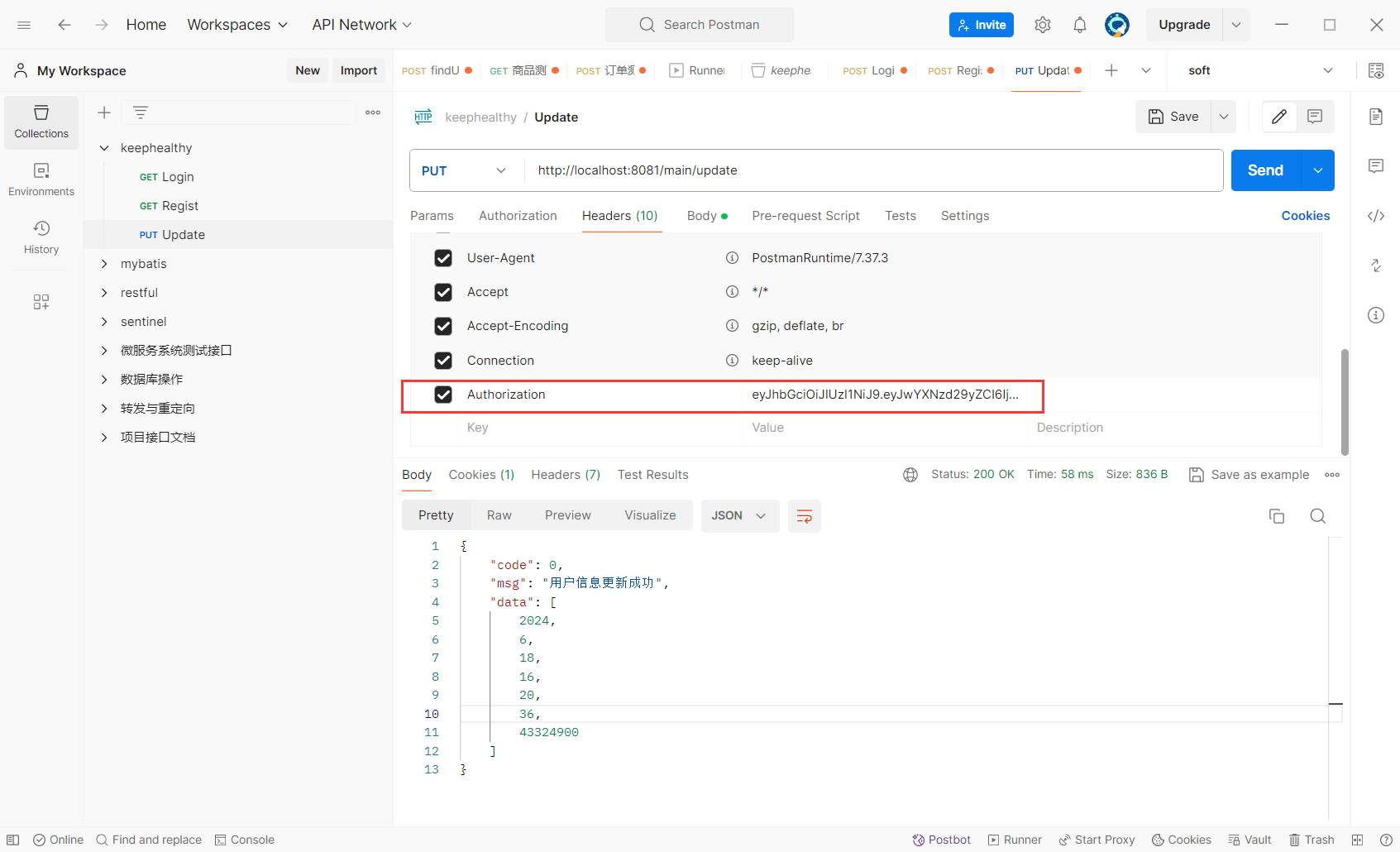
**Login参数复制下来（在下图中位置）**



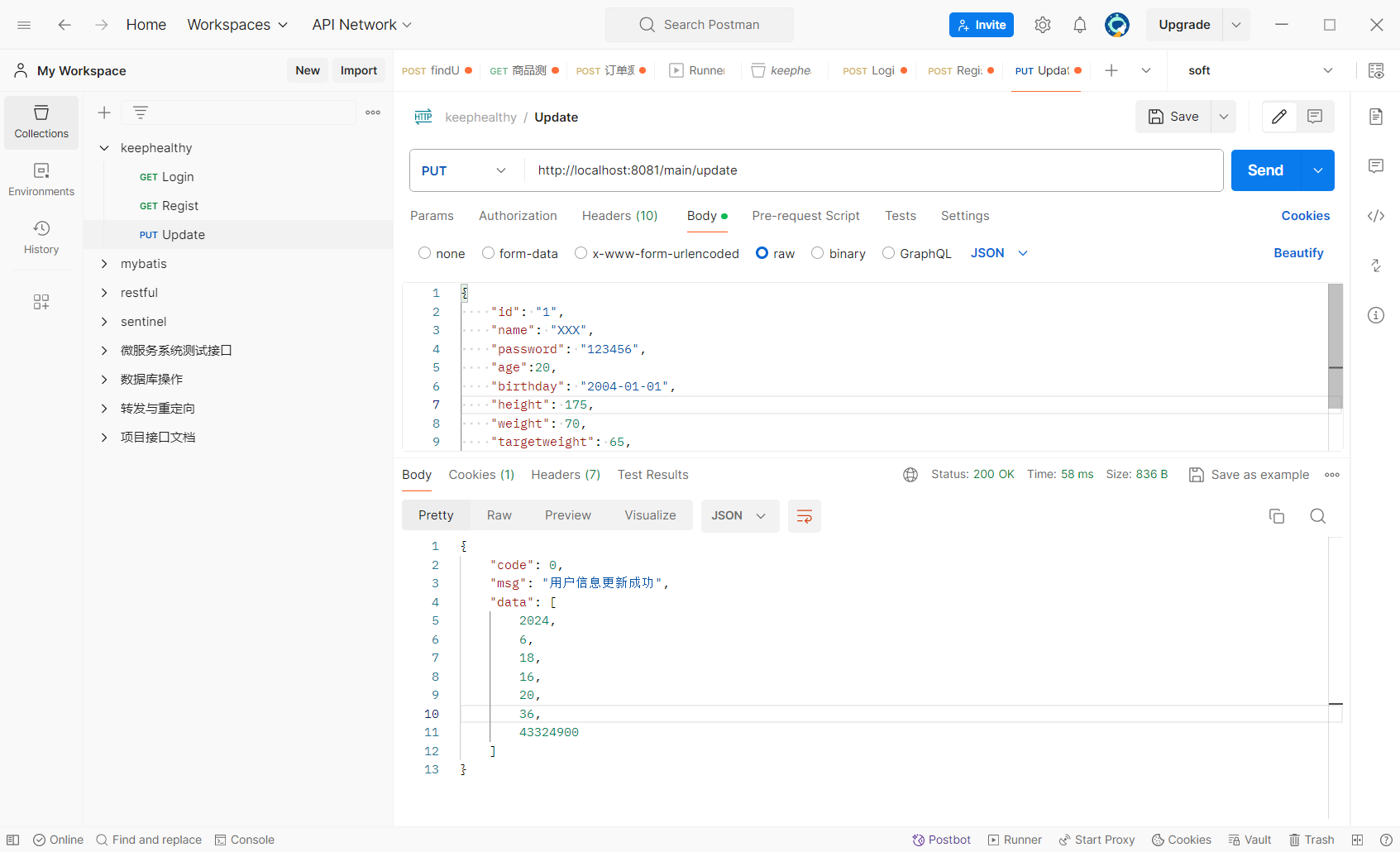
在update的header中

添加一个名为：Authorization的参数

然后把token值放入其中



用户更新成功



**Menu——获取食谱接口测试**

**http://localhost:8081/diet/getMenuRecord**

**例子：**

**menu的sql脚本：**

**CREATE TABLE menu (**

**menuId VARCHAR(50) PRIMARY KEY,**

**make VARCHAR(255),**

**image VARCHAR(255),**

**tips VARCHAR(255),**

**calorie DOUBLE**

**);**

**INSERT INTO menu (menuId, make, image, tips, calorie) VALUES**

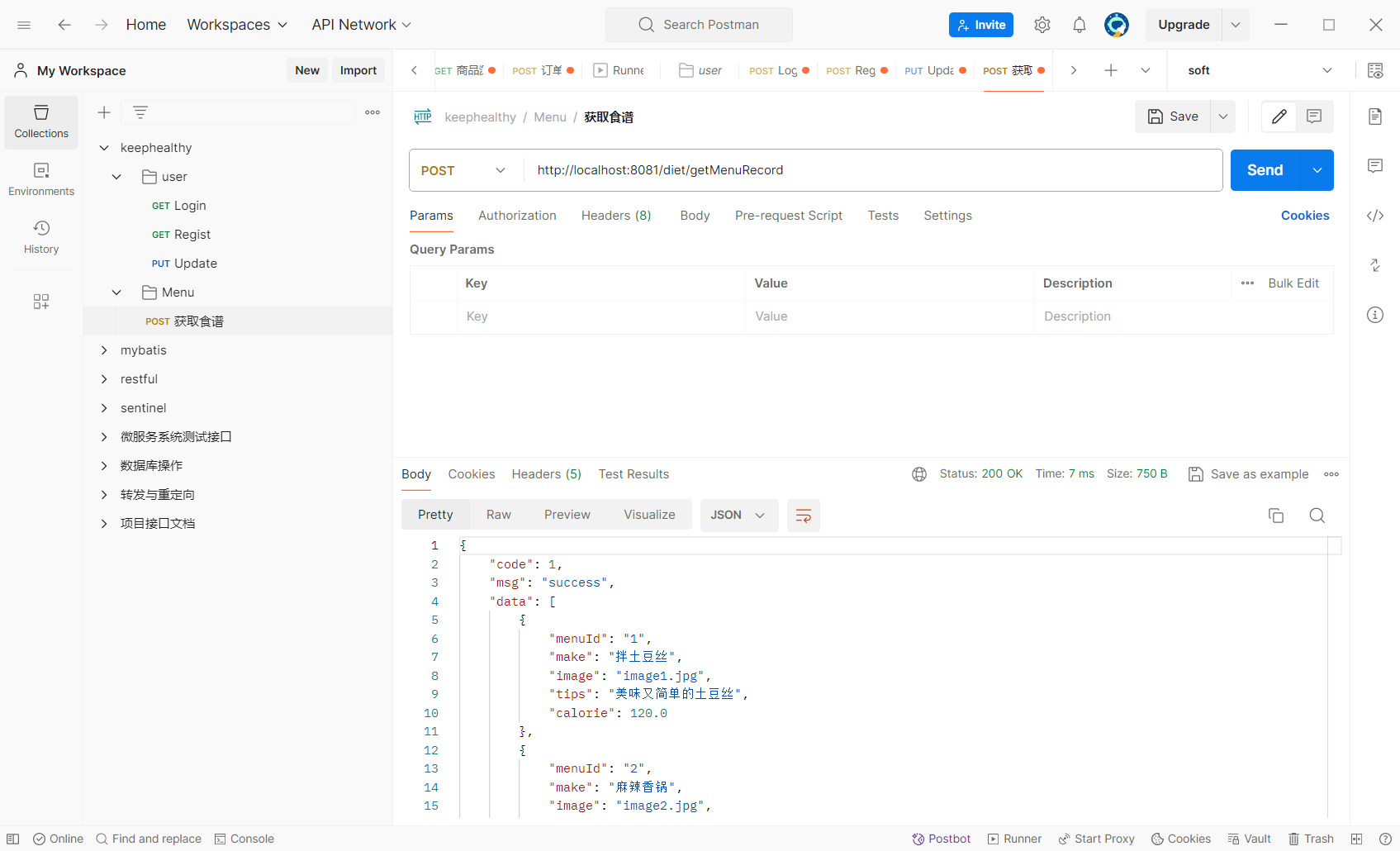
**('1', '拌土豆丝', 'image1.jpg', '美味又简单的土豆丝', 120.0),**

**('2', '麻辣香锅', 'image2.jpg', '香辣爽口的川菜经典', 350.0),**

**('3', '清蒸鱼', 'image3.jpg', '健康又营养的清蒸鱼', 180.0),**

**('4', '凉拌黄瓜', 'image4.jpg', '清凉开胃的夏日小食', 50.0),**

**('5', '牛肉面', 'image5.jpg', '充满回忆的正宗牛肉面', 300.0);**



Sport——获取运动类型接口测试

**http://localhost:8081/sport/type/getSportType**

**sql：**

**CREATE TABLE sport\_type (**

**sport\_id INT PRIMARY KEY,**

**sport\_name VARCHAR(255),**

**single\_calorie INT**

**);**

**INSERT INTO sport\_type (sport\_id, sport\_name, single\_calorie) VALUES**

**(1, '跑步', 100),**

**(2, '游泳', 150),**

**(3, '篮球', 120);**

