**بسم الله الرحمن الرحیم**



**دانشگاه اصفهان**

**دانشکده مهندسی کامپیوتر**

**درس:**

**مهندسی نرم‌افزار**

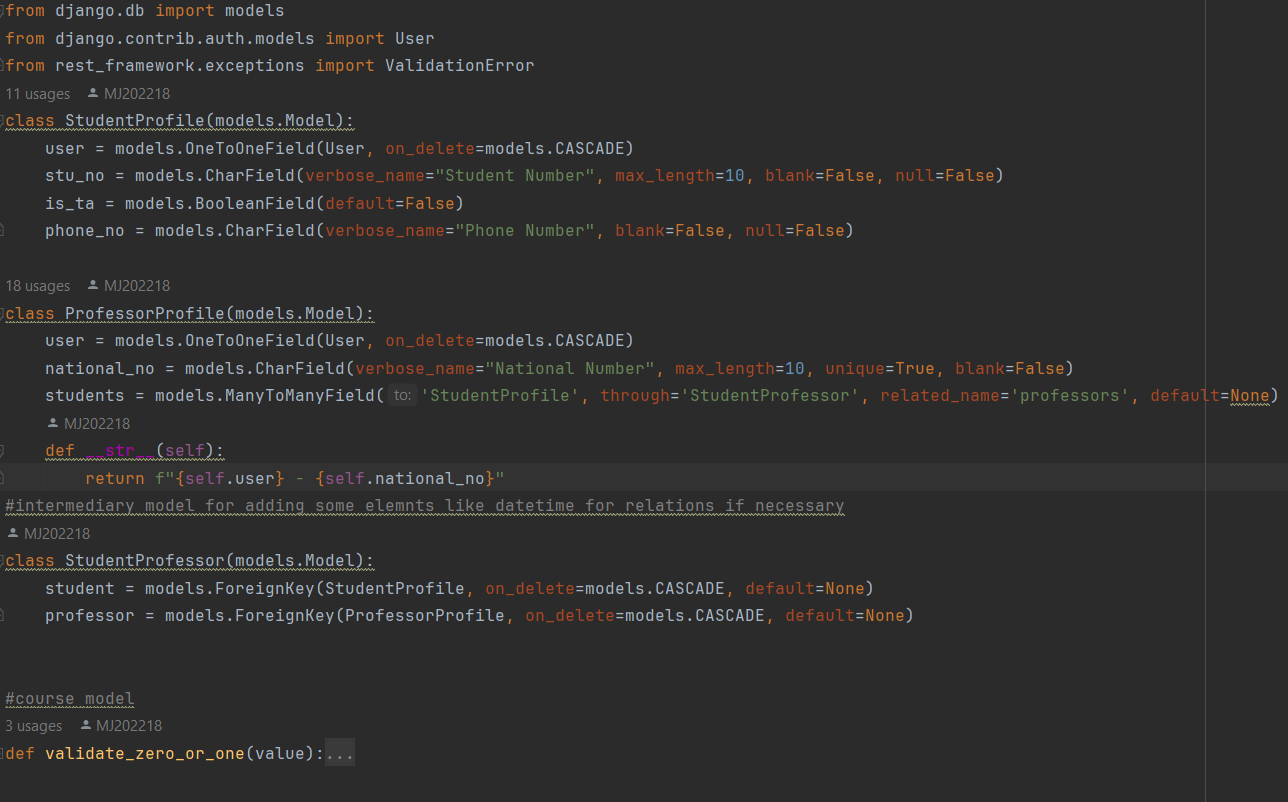
**استاد:**

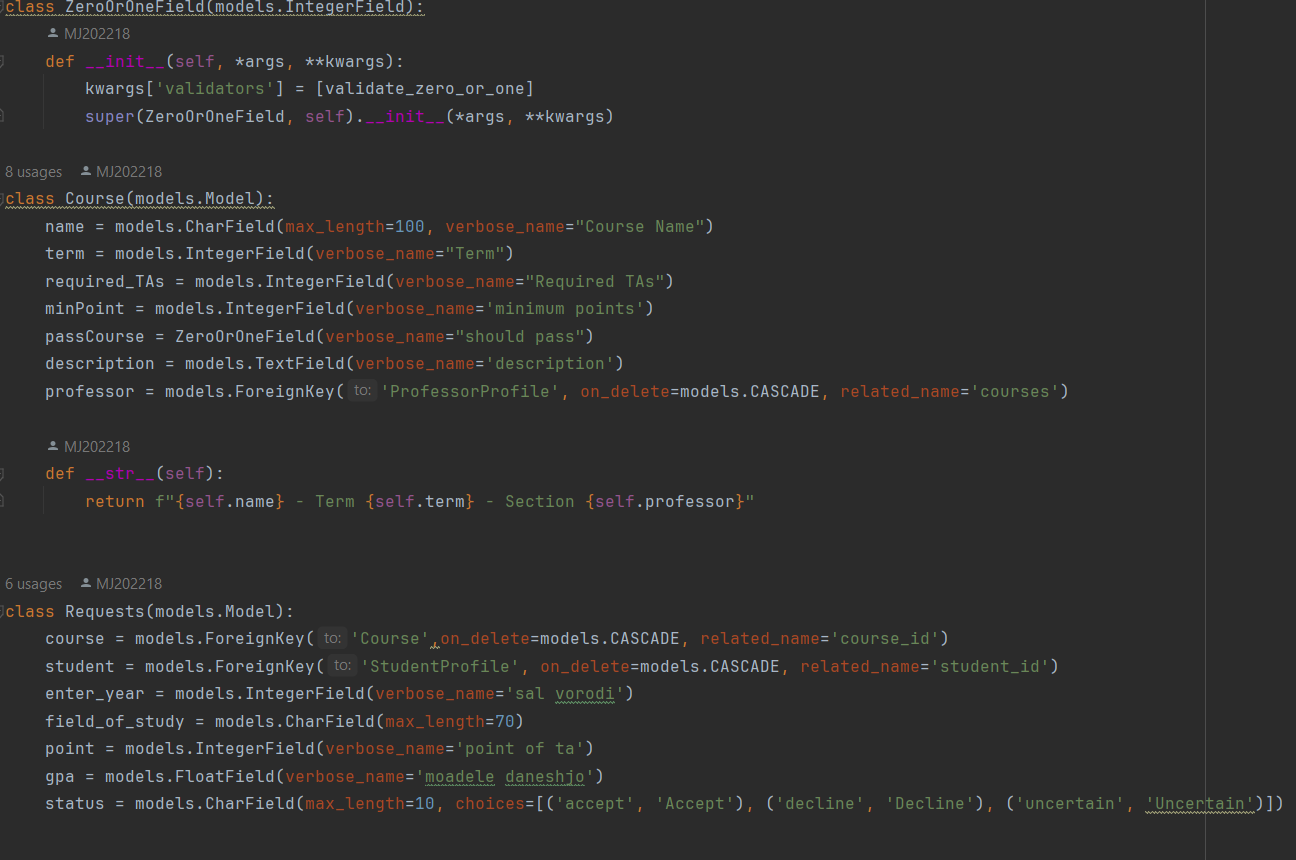
**دکتر فاطمی**

**اعضای گروه:**

**محمد مهدی کتابچی، محمد جعفری، علی حسینی‌فرد، امین کیانی، متین عصب‌الظهور، آرمان خلیلی، یزدان افرا**

مدل های دیتابیسی کنونی:

****

****

**APIها:**

Login

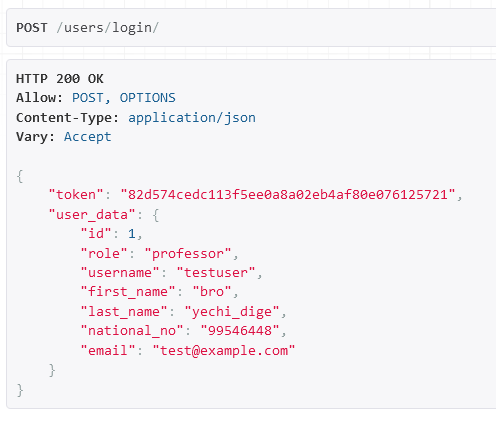
:URL <http://127.0.0.1:8000/users/login/>

method : POST

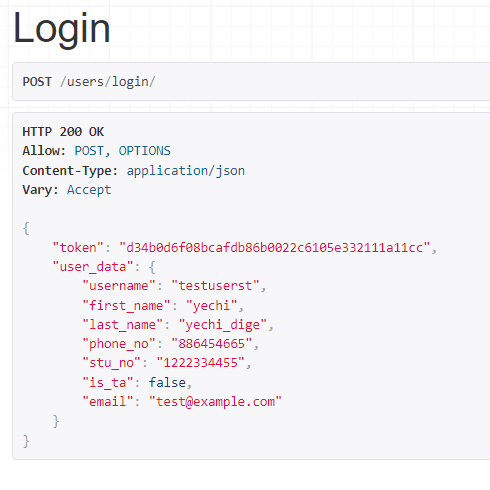
ورودی های الزامی :



خروجی: در حالت صحیح کد 200 در حالت هر حالت نادرست 400

خروجی درحالی که کاربر استاد باشد:  


از قسمت role آن میتوان متوجه پروفسور بودن آن شد همچنین مقدار id متعلق به یوزری است که در پروفایل استاد استفاده شده. این در قسمت های بعدی استفاده زیادی خواهد داشت.

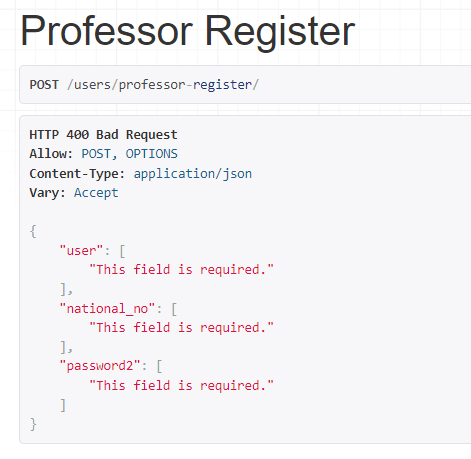
خروجی درحالی که کاربر دانشجو باشد:

Professor register(sign-up)

url : <http://127.0.0.1:8000/users/professor-register/>

method : POST

ورودی های الزامی :



ورودی های اصلی(ورودی های که به طور عادی انتظار انها را داریم) :

{

"user": {

"username": "testuser",

"first\_name": "yechi",

"last\_name": "yechi\_dige",

"email": "test@example.com",

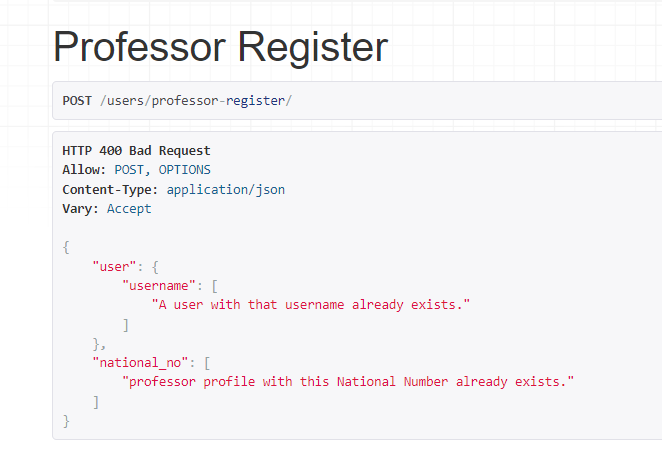
"password": "testpassword"

},

"national\_no": "1222334455",

"password2": "testpassword"

}

خروجی در صورت اطلاعات تکراری یا غلط:  


خروجی در صورت درخواست صحیح (برگرداندن اطلاعات کاربری که ثبت شده)



Student register(sign-up)

url : <http://127.0.0.1:8000/users/student-register/>

method : POST

ورودی های الزامی :



ورودی های اصلی(ورودی های که به طور عادی انتظار انها را داریم) :

{

"user": {

"username": "testuser9",

"first\_name": "yechi",

"last\_name": "yechi\_dige",

"email": "test@example.com",

"password": "testpassword"

},

"stu\_no": "1222334455",

"phone\_no": "0991199853",

"password2": "testpassword"

}

خروجی صحیح:



خروجی در صورت اطلاعات غلط شبیه قبلی

Logout

url : <http://127.0.0.1:8000/users/logout/>

method :GET

ورودی های الزامی :

بدون اینکه وارد شده باشیم اجازه دسترسی نداریم و در ریکوئست گت باید توکن موجود باشد.



ورودی صحیح:

Authorization: token b2d617908123bd501e62f9bc8c81e3301e7e4aa5

باید هدری با این مشخصات به طوری که توکن بعد عبارت token باید با یک فاصله نوشته شود پاس داده شود

خروجی با اطلاعات غلط:

شبیه عکس ورودی های الزامی  
  
خروجی با اطلاعات صحیح :

تنها کد 200 برگردانده میشود.

ProfessorCourseAPIView

url : http

://127.0.0.1:8000/users/professor/get-lesson/

method :GET

درس‌های مربوط به هر استاد داده می‌شوند در صورت احراز هویت شده بودن.

ورودی های الزامی :

بدون اینکه وارد شده باشیم اجازه دسترسی نداریم و در ریکوئست گت باید توکن موجود باشد  
  
مشابه عکس ورودی الزامی Logout

ورودی صحیح:

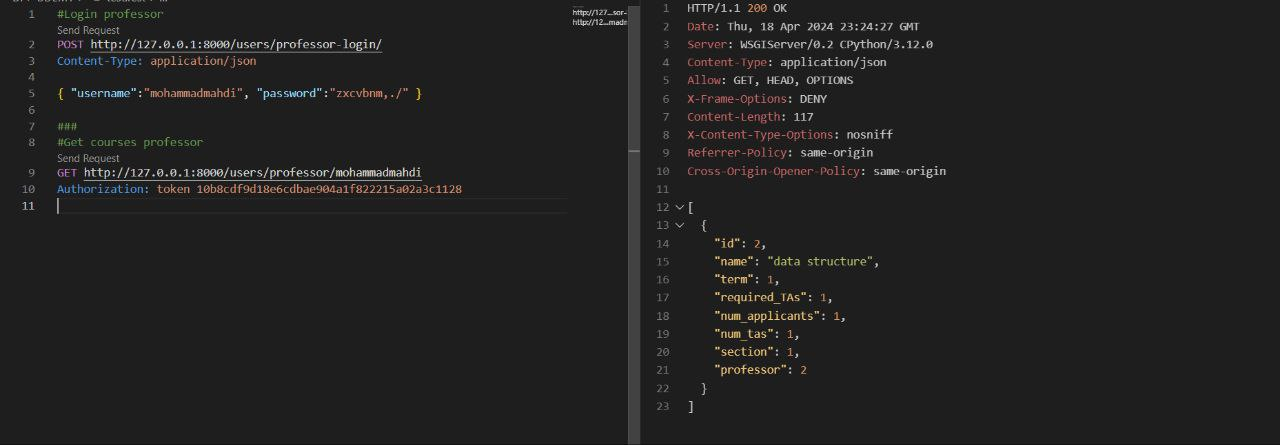
Authorization: token b2d617908123bd501e62f9bc8c81e3301e7e4aa5

باید هدری با این مشخصات به طوری که توکن بعد عبارت token باید با یک فاصله نوشته شود پاس داده شود

خروجی با اطلاعات غلط:

شبیه عکس ورودی های الزامی logout  
  
خروجی با اطلاعات صحیح :

کد 200 برگردانده میشود. همراه لیستی از دروس

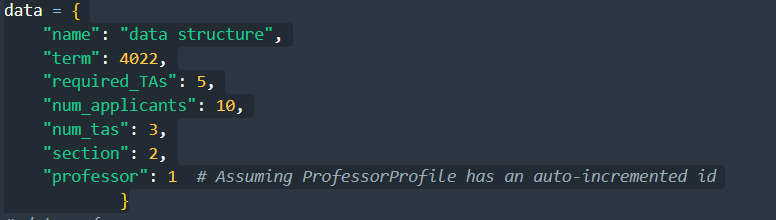


CourseRegisterView

Method: post  
  
url : http

://127.0.0.1:8000/users/ professor/create-lesson/

ورودی ها:



**فیلد پروفسور همان آیدی ای را می گیرد که در قسمت لاگین بر می گردانیم**.

خروجی:



جای name اسم درس قرار می گیرید

نکته:

اسم درس دوبار نمی تواند تکرار شود.

ProfessorDetailsView  
method:get  
url : http://127.0.0.1:8000/users/ professor/get-detail/<int:professor\_id>

ورودی:

شماره آیدی پروفسور (گرفته شده در لاگین)

خروجی:



لیست دانشجو های اول دانشجو هایی هستند که متعلق به استادند. و لیست دانشجو های دوم ریز اطلاعات مربوط به آن دانشجویان.

StudentProfilePartialUpdateView  
method:patch

url : http://127.0.0.1:8000/users/student/profile/<int:id>/

اطلاعات دانشجویان به طور تکه ای آپدیت میکند یعنی لازم نیست حتما کل آن یکجا آپدیت شود قسمت های جدا هم اپدیت می شود. فقط ویژگی ها خاص دانشجو مثل شماره تلفن و شماره دانشجویی و تی ای بودن می تواند آپدیت شود. اسم و فامیل و ایمیل و ... با آپدیت یوزر آپدیت میشود

ورودی:

مثلا میخوام فقط stu\_no بودن رو آپدیت کنم.

data = {

    'stu\_no':'99546448',

}

اگر چیزای دیگه خواستم اونا رو هم اضافه می کنم. همچنین ای دی همونی هست که از لاگین گرفتیم.(مال یوزره)

خروجی:

اطلاعات جدید دانشجو

ProfessorProfilePartialUpdateView  
method:patch

url :http://127.0.0.1:8000/users/student/profile/<int:id>/

ورودی

شبیه آپدیت دانشجو یعنی مثلا چیزی مثل کدملی رو آپدیت می کنی نه اسم و فامیل چون اون مال آپدیت یوزر می شه.

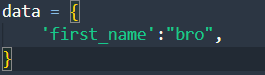
خروجی

شبیه آپدیت دانشجو

UserPartialUpdateView  
method:patch

url :http://127.0.0.1:8000/users/ update/<int:id>/

ورودی



ویژگی های یوزر به طور بخش بخش یا کامل آپدیت می شود. آیدی هم همان آیدی ای است که موقع لاگین میگیریم.

خروجی

اطلاعات جدید یوزر

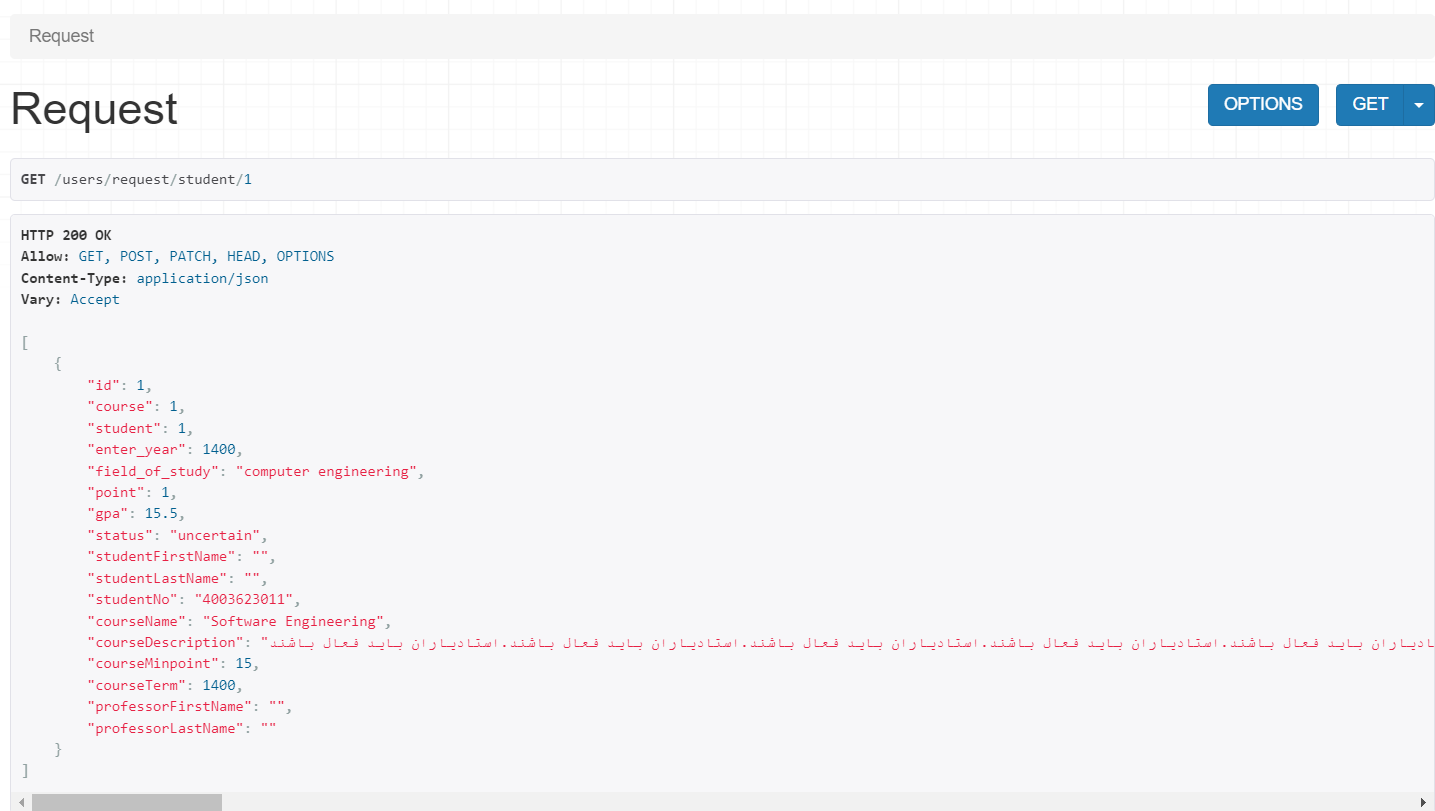
CourseDeleteView  
method:delete

url :http://127.0.0.1:8000/users/ professor/delete-lesson/<str:name>/

ورودی:

اسم درس را برای ورودی باید داده شود.

خروجی :ندارد

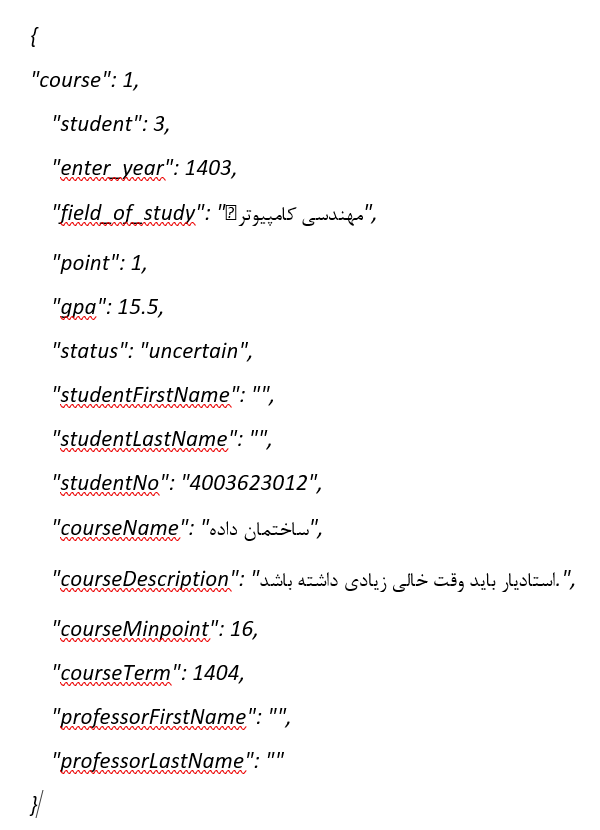
در اینجا با روش اگر استاد وارد شود اطلاعات درخواست‌های مرتبط با استاد نمایش داده می‌شود و اگر دانشجو وارد شود اطلاعات درخواست‌های دانشجو نمایش داده می‌شود.

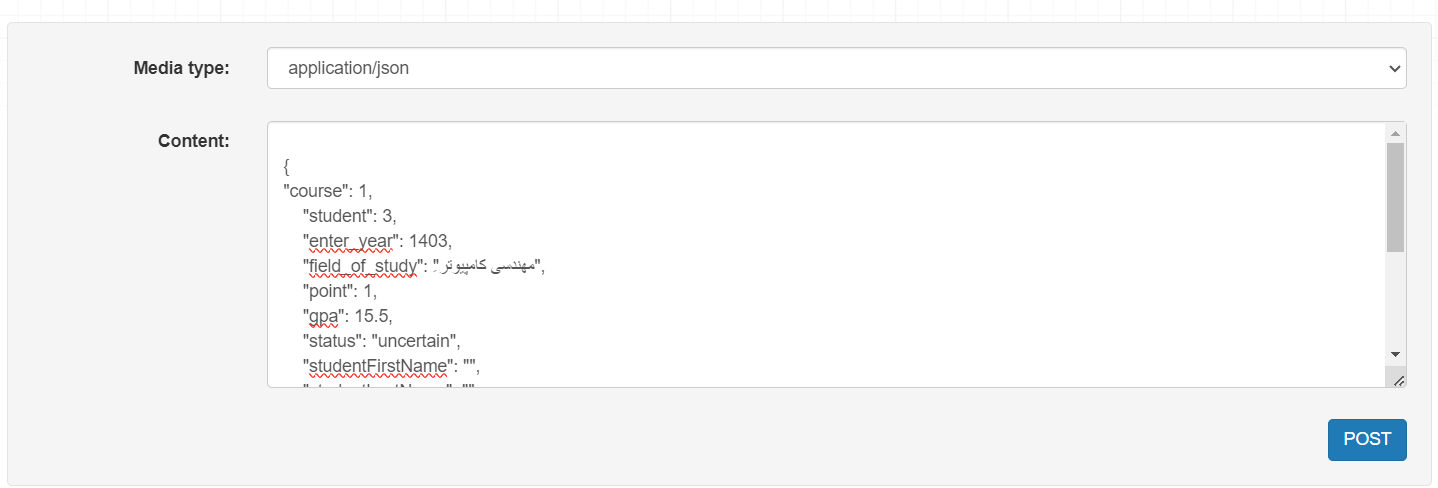
خروجی برای کاربر دانشجو

خروجی برای کاربر استاد

همچنین با روش هم از سمت استاد و هم از سمت دانشجو امکان ذخیره درخواست‌ها فراهم می‌شود.

برای مثال فرض کنیم میخواهیم درخواست زیر را کنیم:

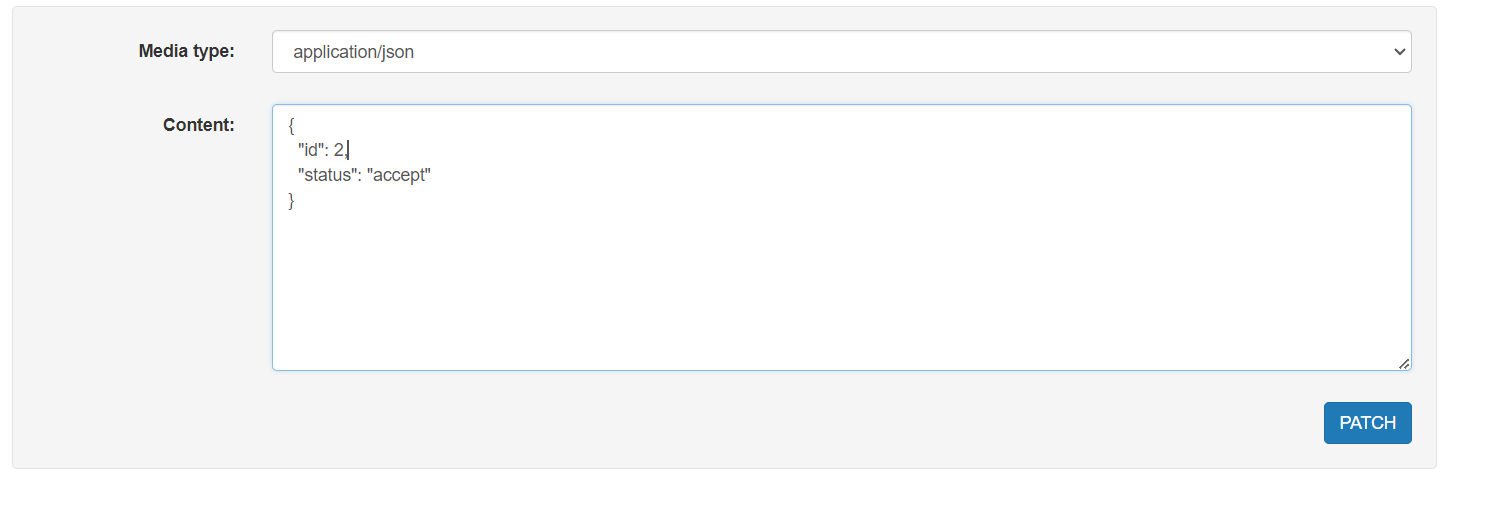


برای اینکار این اطلاعات را به صورت جیسون در محیطی که فراهم کرده می‌نویسیم و سپس روی دکمه کلیک می‌کنیم:

*نتیجه به صورت زیر است:*



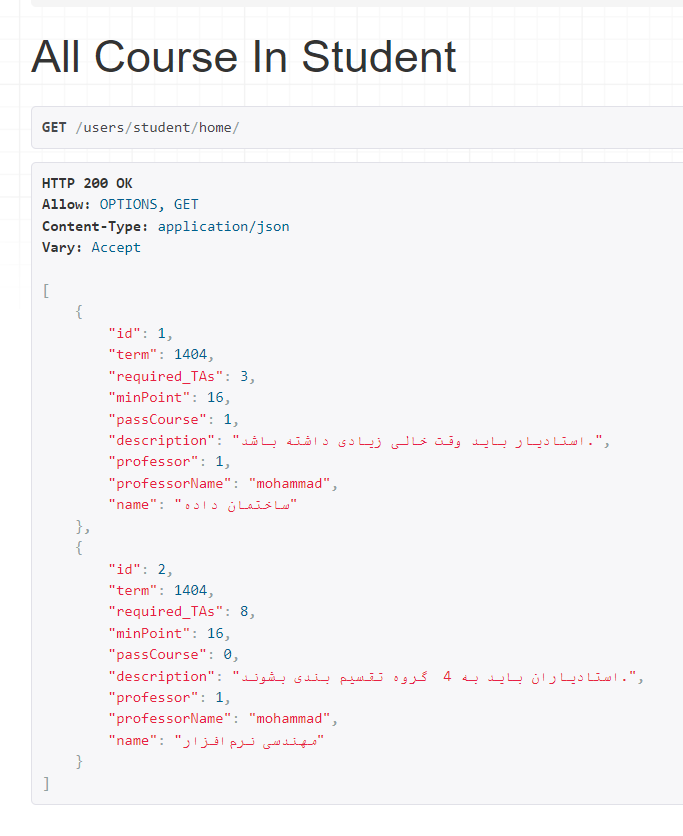
*با استفاده از روش می‌توانیم وضعیت درخواست‌ها را عوض کنیم. برای مثال اگر بخواهیم درخواست 2 را قبول کنیم به صورت زیر عمل می‌کنیم:*

**

نتیجه به صورت زیر می‌شود:

در اینجا با روش تمام دروس برای دانشجویانی که برای آنها توکن تشکیل شده است (لاگین کرده‌اند) نمایش داده می‌شود.

**خروجی:**



**کد:**

from rest\_framework import authentication  
class BearerAuthentication(authentication.TokenAuthentication):  
 *'''  
 Simple token based authentication using utvsapitoken.  
 Clients should authenticate by passing the token key in the 'Authorization'  
 HTTP header, prepended with the string 'Bearer '. For example:  
 Authorization: Bearer 956e252a-513c-48c5-92dd-bfddc364e812  
 '''* keyword = ['token','bearer']  
 def authenticate(self, request):  
 auth = authentication.get\_authorization\_header(request).split()  
 if not auth:  
 return None  
 if auth[0].lower().decode() not in self.keyword:  
 return None  
  
 if len(auth) == 1:  
 msg =\_('Invalid token header. No credentials provided.')  
 raise authentication.exceptions.AuthenticationFailed(msg)  
 elif len(auth) > 2:  
 msg =\_('Invalid token header. Token string should not contain spaces.')  
 raise authentication.exceptions.AuthenticationFailed(msg)  
 try:  
 token = auth[1].decode()  
 except UnicodeError:  
 msg =\_('Invalid token header. Token string should not contain invalid characters.')  
 raise authentication.TokenAuthentication.exceptions.AuthenticationFailed(msg)  
 return self.authenticate\_credentials(token)

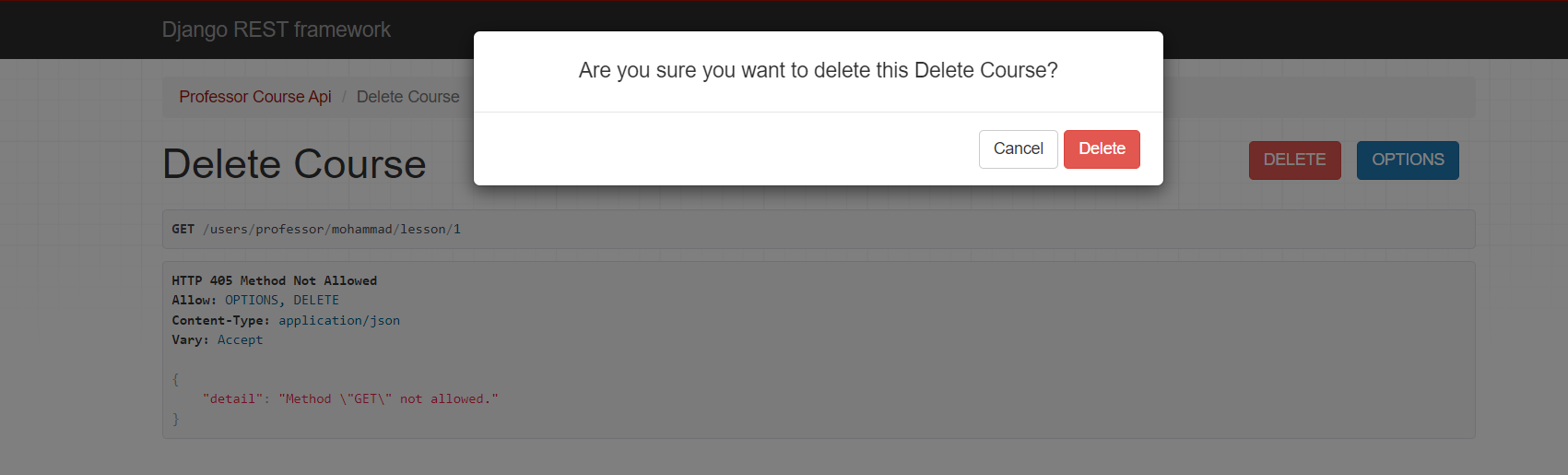
کد بالا برای احراز هویت با استفاده از توکن‌های نوع و مورد استفاده قرار می‌گیرد. این کلاس از کلاس پایه از کتابخانه استفاده می‌کند.

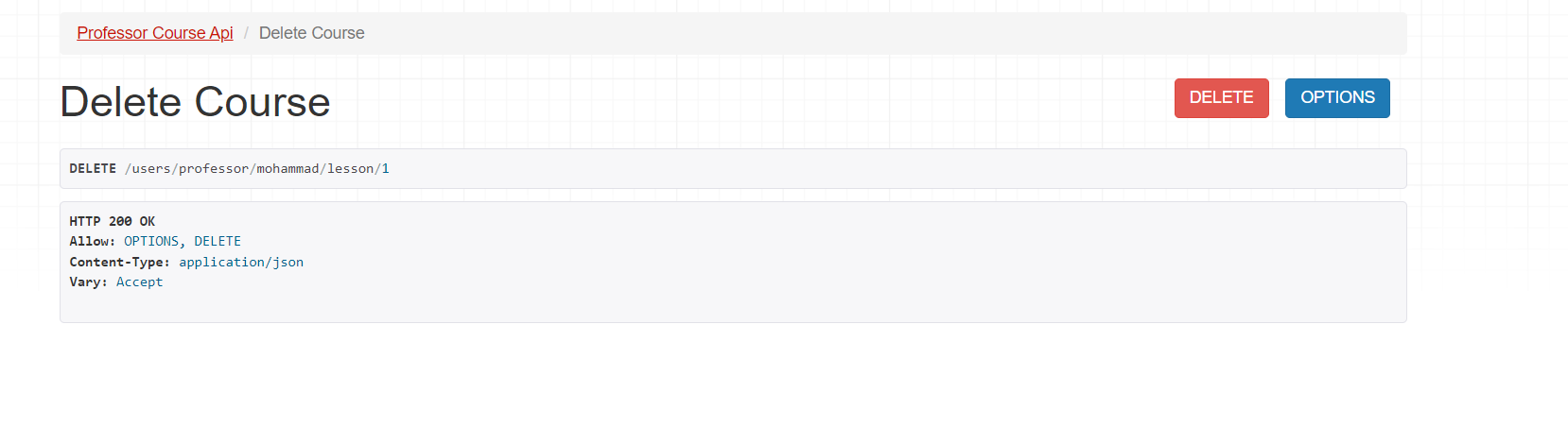
*deleteCourse*

*DELETE*

*در اینجا با روش با استفاده از اسم استاد و آیدی درس می‌توانیم درس را حذف کنیم.*

*با کلیک روی درس مورد نظر با آیدی وارد شده حذف می‌شود.*

**

*نتیجه:*