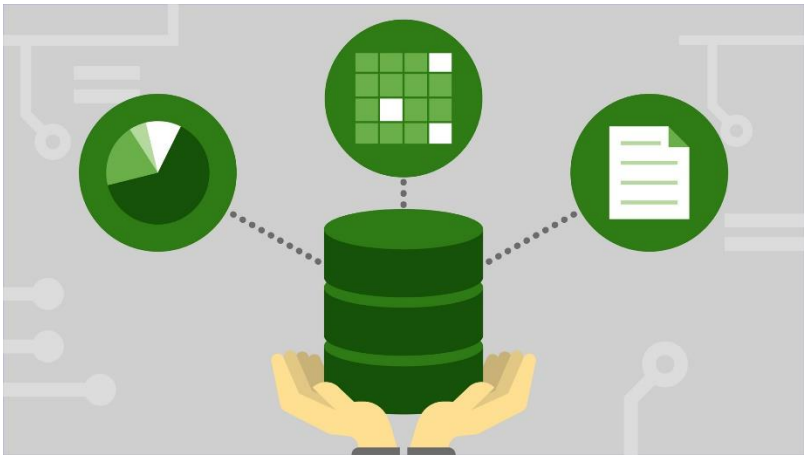
به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر



# آزمایشگاه پایگاه‌داده

## دستورکار شماره 2

شماره دانشجویی

810196443

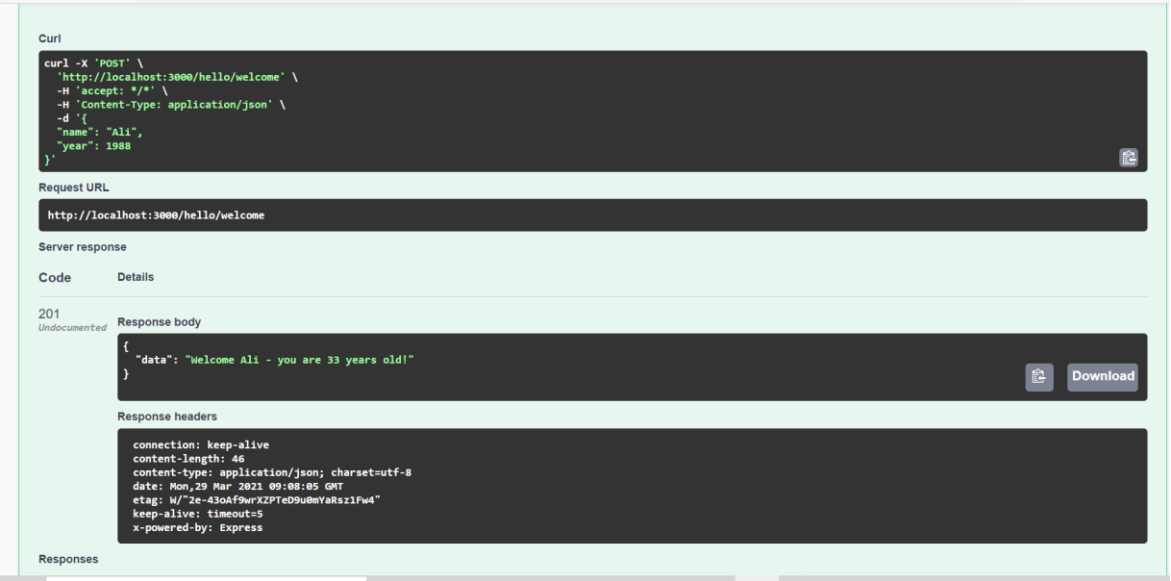
فروردین ۱۴۰۰

هومان چمنی

# گزارش فعالیت‌های انجام شده

## بخش اول:

این بخش توضیح خاصی نداشته و صرفا مراحل گفته شده در بخش مقدمه به ترتیب اجرا شده است. عکس زیر نتیجه‌ای از صفحه Swagger بخش مقدمه بوده که یک درخواست از نوع Post ارسال شده است و نتیجه آن قابل مشاهده است.

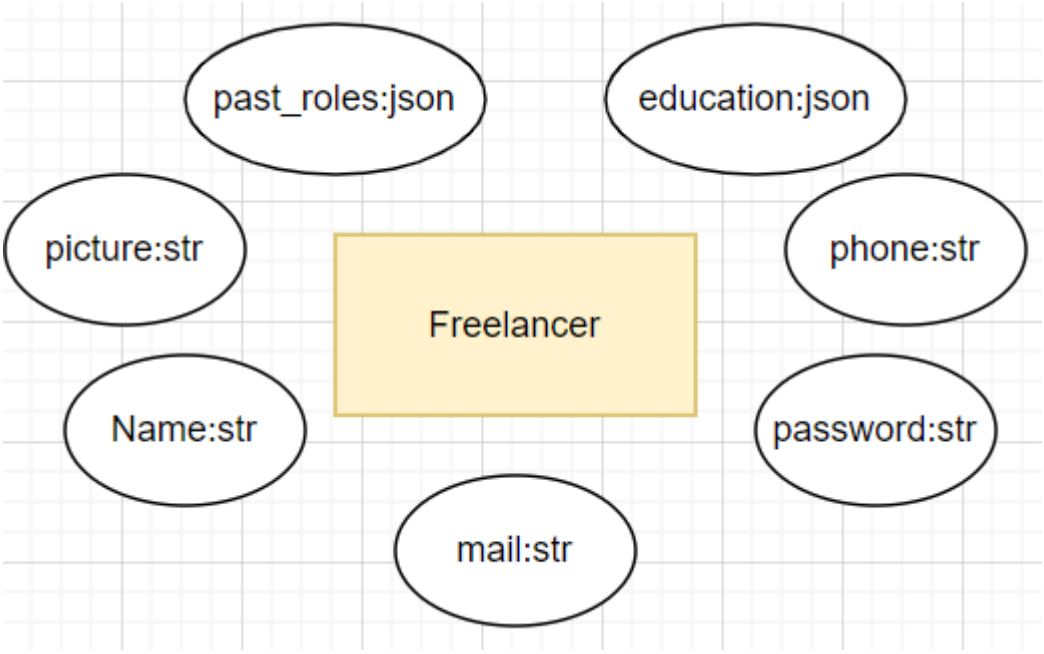



همچین آدرس مخزن آزمایش ۲ پایگاه‌داده به صورت repo link بوده و شناسه کامیت که مربوط به اتمام بخش اول از دستور کار می‌باشد نیز به صورت bbc9e856b42851ced475328dd568c7fc703d064e می‌باشد.

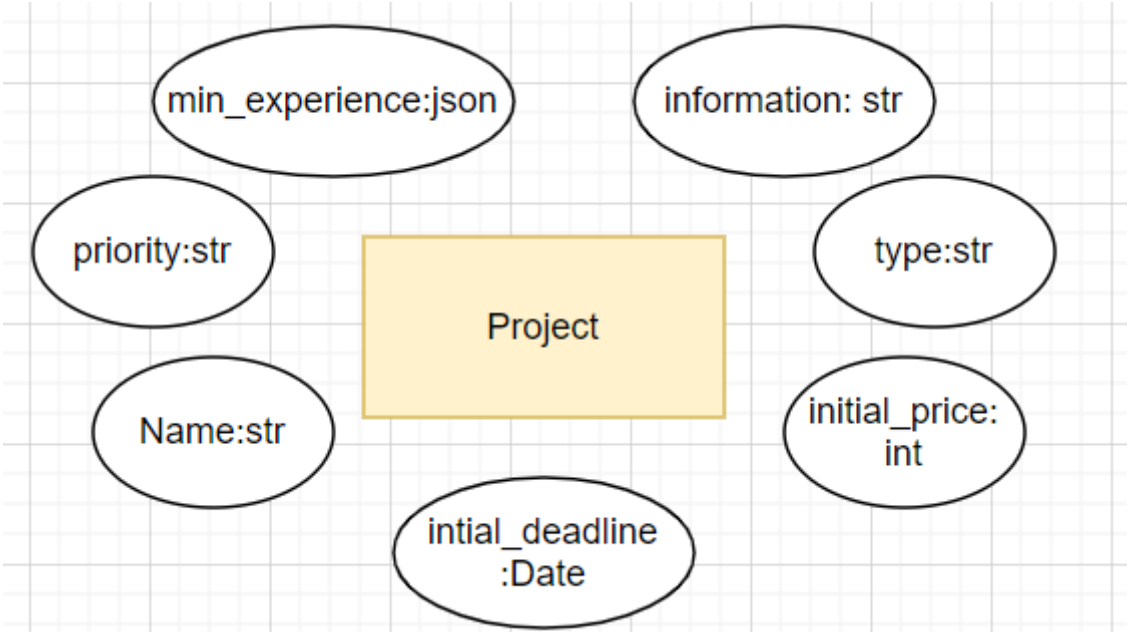
## بخش دوم:

قابلیت‌های در نظر گرفته شده:

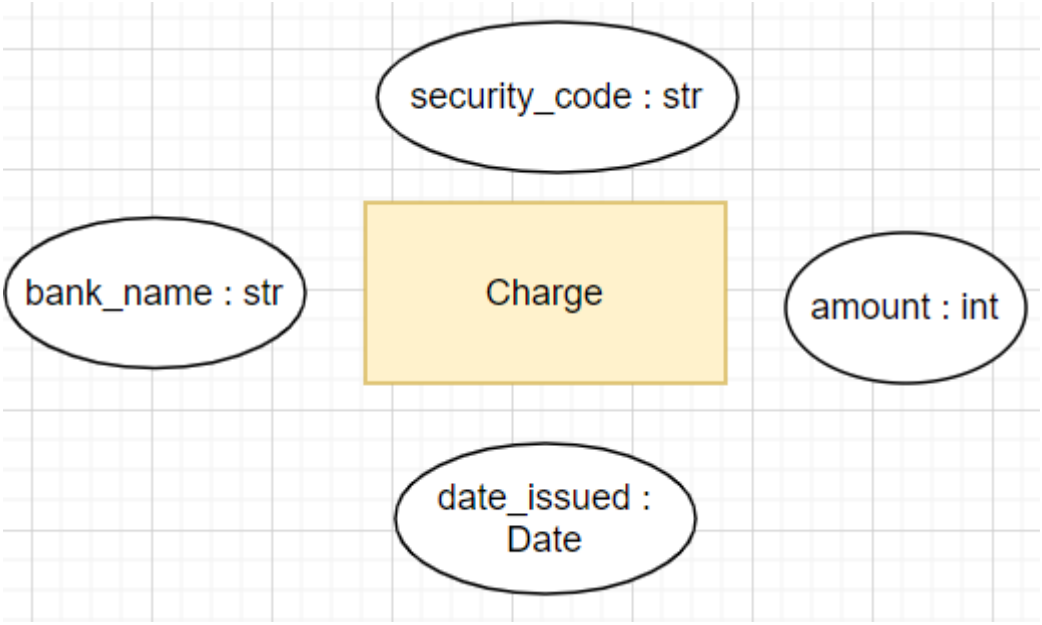
| ردیف | کاربر | امکانات | متدها |
|---|---|---|---|
| ۱ | کارفرما | کارفرما می‌تواند سفارش یک پروژه را ثبت و آنرا ویرایش کند. | Get – Post – Put - Delete |
| ۲ | فریلنسر | فریلنسر می‌تواند ثبت‌نام کرده و رزومه خود را بروزرسانی کند. | Get – Post – Put - Delete |
| ۳ | فریلنسر | فریلنسر می‌تواند پروژه‌های کاری را جستجو و برای آنان درخواست ارسال کند. | Get – Post – Put - Delete |
| ۴ | کارفرما | کارفرما می‌تواند از طریق درگاه پرداخت بانکی اکانت خود را شارژ کرده تا هزینه پروژه ها از آن راه تامین شود. | Get – Post – Put - Delete |



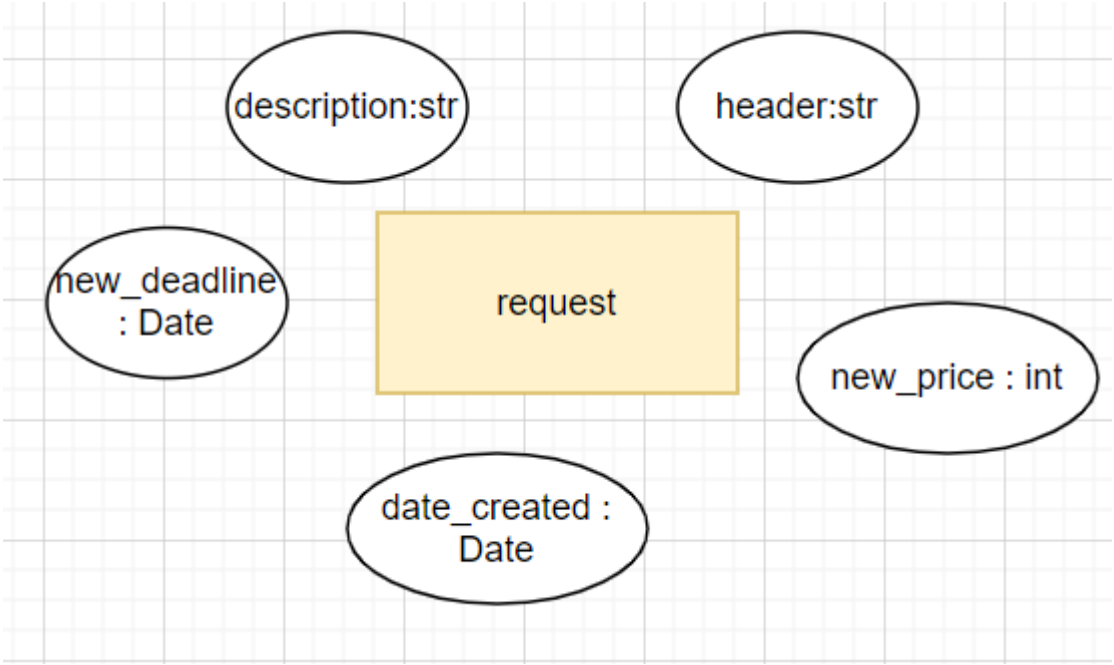

| قابلیت | متدها | توضیح و ورودی | خروجی‌ها |
|---|---|---|---|
| ثبت‌نام فریلنسر در سیستم /freelancer | Get | گرفتن اطلاعات یک فریلنسر با استفاده از fid که به صورت freelancer/fid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن اطلاعات freelancer مذکور<br>کد ۴۰۴ : (خطا) یافته نشدن freelancer مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Post | تعریف یک فریلنسر جدید با استفاده از خود فریلنسر که به صورت freelancer/ درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۰ : دسترسی امکان‌پذیر نمی‌باشد |
| | Put | ادیت اطلاعات یک فریلنسر با استفاده از fid که به صورت freelancer/fid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن freelancer مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Delete | حذف یک فریلنسر با استفاده از fid که به صورت freelancer/fid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن freelancer مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |




| قابلیت | متدها | توضیح و ورودی | خروجی‌ها |
|---|---|---|---|
| ثبت پروژه در سیستم توسط کارفرما /taskmaster/{tid} | Get | گرفتن اطلاعات یک پروژه با استفاده از pid و tid که به صورت tid/projects/pid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن اطلاعات project مذکور<br>کد ۴۰۴ : (خطا) یافته نشدن project مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Post | تعریف یک پروژه جدید با استفاده از آیدی کارفرما (tid) و اطلاعات پروژه جدید که به صورت tid/projects درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۰ : (خطا) دسترسی امکان‌پذیر نمی‌باشد<br>کد ۴۰۴ : taskmaster مورد نظر وجود ندارد |
| | Put | ادیت اطلاعات یک پروژه با استفاده از pid و tid و داده جدید صورت tid/projects/pid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن project مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Delete | حذف یک پروژه با استفاده از pid و tid که به صورت tid/projects/pid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن project مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |



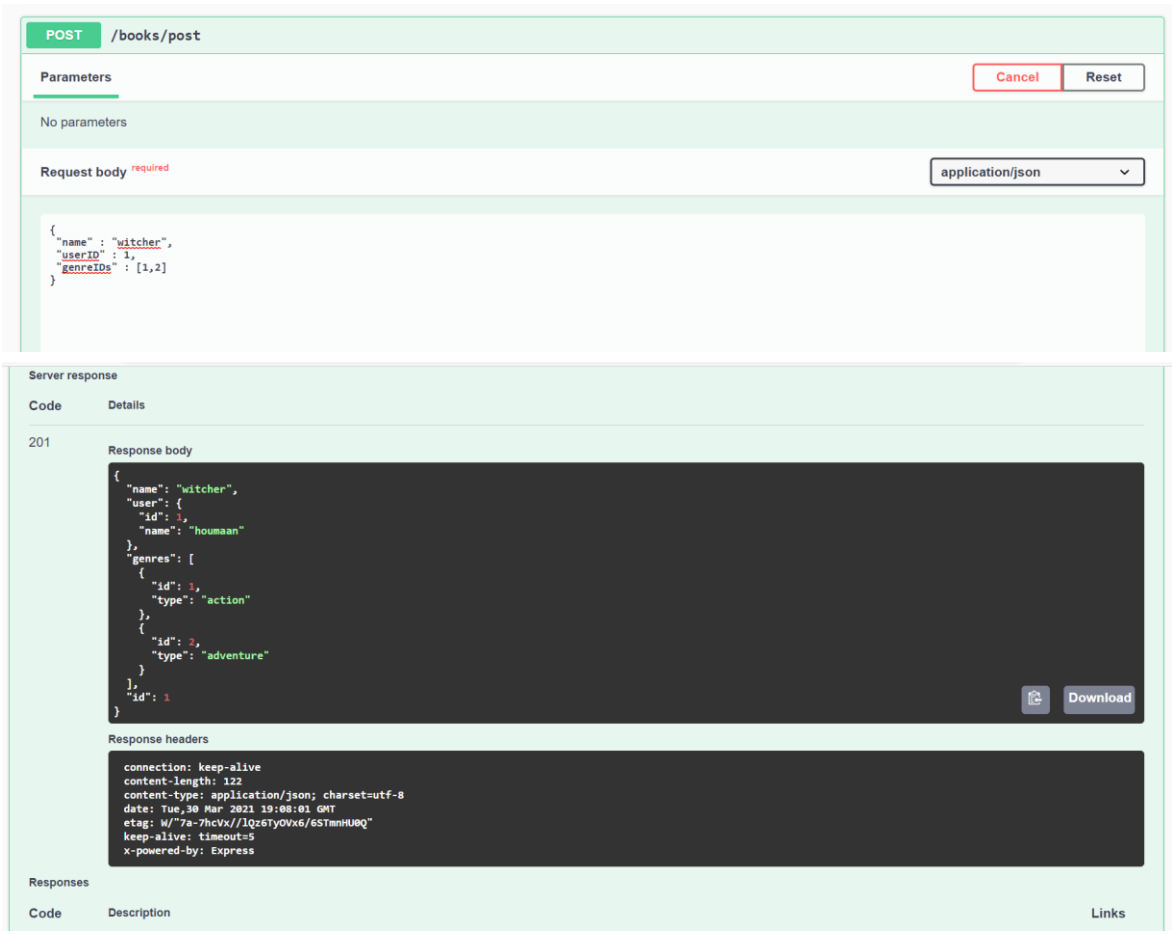

| قابلیت | متدها | توضیح و ورودی | خروجی‌ها |
|---|---|---|---|
| شارژ اکانت توسط کارفرما /taskmaster/{tid} | Get | گرفتن اطلاعات یک تراکنش با استفاده از cid و tid که به صورت tid/charges/cid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن اطلاعات تراکنش مذکور<br>کد ۴۰۴ : (خطا) یافته نشدن تراکنش مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Post | ایجاد یک تراکنش جدید با استفاده از آیدی کارفرما (tid) و اطلاعات تراکنش جدید که به صورت tid/charges درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۰ : (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Put | ادیت اطلاعات یک تراکنش با استفاده از cid و tid و داده جدید صورت tid/charges/cid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن تراکنش مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Delete | حذف یک تراکنش با استفاده از cid و tid که به صورت tid/charges/cid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن تراکنش مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |



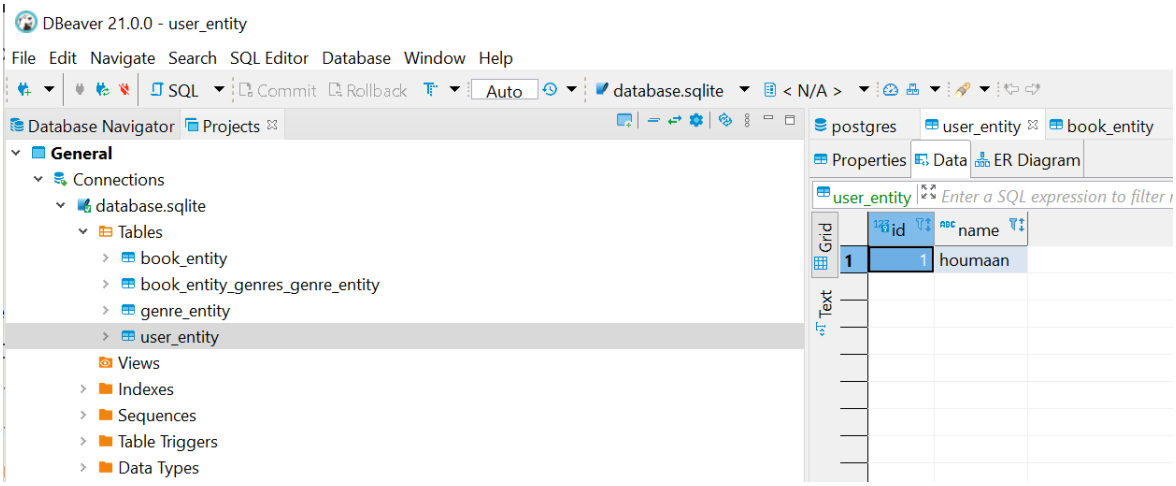

| قابلیت | متدها | توضیح و ورودی | خروجی‌ها |
|---|---|---|---|
| ارسال درخواست توسط فریلنسر /freelancer/{fid} | Get | گرفتن اطلاعات یک درخواست با استفاده از rid و fid که به صورت fid/reqs/rid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن اطلاعات request مذکور<br>کد ۴۰۴ : (خطا) یافته نشدن request مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Post | ایجاد یک درخواست جدید با استفاده از آیدی فریلنسر (fid) و اطلاعات درخواست جدید که به صورت fid/reqs درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۰ : (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Put | ادیت اطلاعات یک درخواست با استفاده از rid و fid و داده جدید صورت fid/reqs/rid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن request مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |
| | Delete | حذف یک درخواست با استفاده از rid و fid که به صورت fid/reqs/rid درخواست زده می‌شود. | کد ۲۰۰ : برگرداندن پیغام موفقیت<br>کد ۴۰۴ : (خطا) یافته نشدن request مذکور<br>کد ۴۰۰: (خطا) دسترسی امکان‌پذیر نمی‌باشد |

## بخش سوم:

پس از انجام دادن مراحل طبق توضیحاتی که در سایت مذکور داده شده بود (البته مشکلات متعددی وجود داشت و کد نیاز به دیباگ داشت!) برای تست کردن کد از طریق Swagger ابتدا یک user که نام آن Houmaan است تعریف کرده و آیدی آن ۱ شد. سپس دو ژانر به نام‌های action با آیدی ۱ و adventure با آیدی ۲ تعریف کردم و در آخر نیز یک کتاب با نام Witcher که متعلق به Houmaan بوده و دو ژانر مذکور را دارا می‌باشد با استفاده از متد post تعریف کردم. شکل‌های زیر نتیجه را نشان می‌دهند:



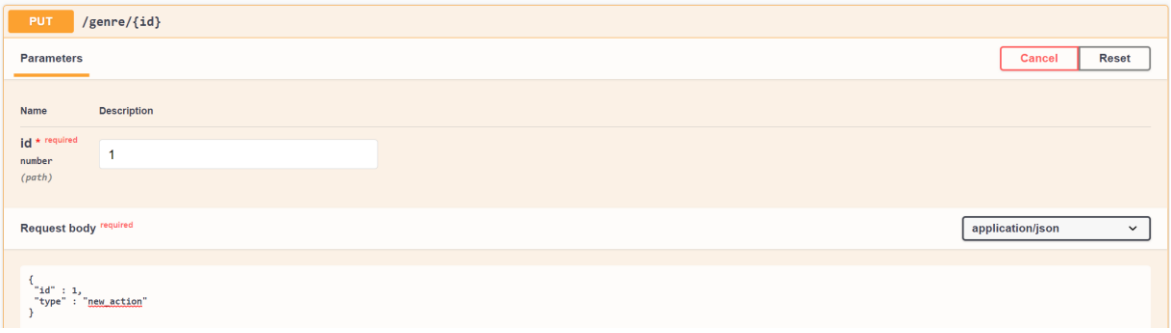

شکل زیر نیز نتیجه باز کردن فایل db.sqlite درون dbeaver می‌باشد که نشان می‌دهد user ما به درستی ساخته شده است:



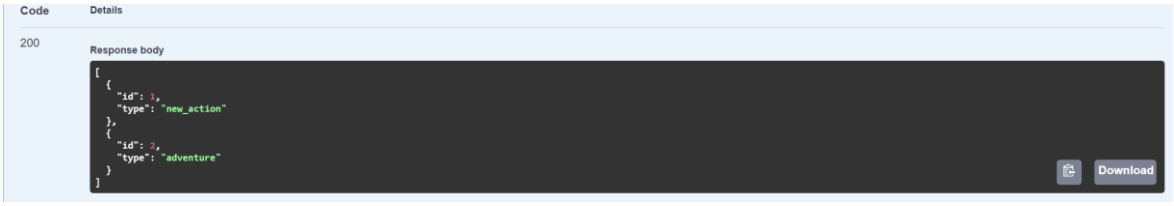
## بخش چهارم:

پس از انجام دادن مراحل اضافه کردن متدهای Put و Delete به سه عنصر books,genre,user و ران کردن دوباره پروژه همان‌طور که در عکس زیر مشاهده می‌شود قابلیت‌های هر سه entity کامل شده‌اند (فایل ormconfig نیز طبق دستور کار آپدیت شده و موارد در دیتابیس postgres نیز بررسی شده‌اند):

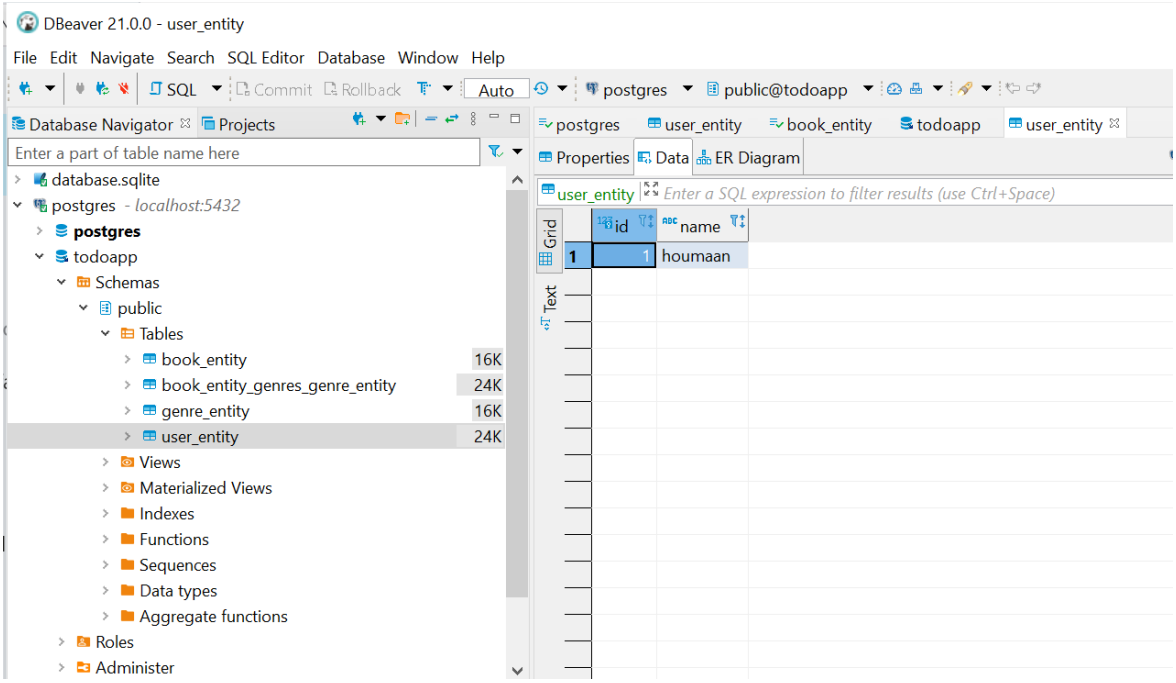

حال با استفاده از متد Put می‌خواهیم اولین ژانر خود را عوض کنیم. در بدنه درخواست آیدی را برابر ۱ قرار داده و تایپ جدید را هم new_action می‌گذاریم که جای تایپ قبلی که action بود (در شکل بخش سوم مشخص است) ثبت شود.




اگر پس از دستور بالا دوباره تمام ژانرها را با دستور Get دریافت کنیم مشخص می‌شود که تایپ ژانر با آیدی ۱ عوض شده است:



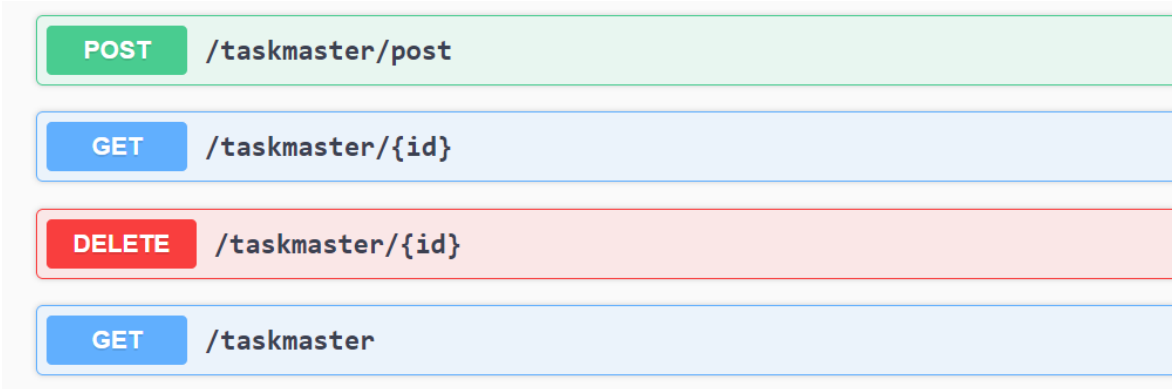
شکل زیر نیز مربوط به Dbeaver می‌باشد که پس از متصل کردن کد به postgres و ساختن یک user مانند قبل به نام houmaan مشاهده می‌شود که این user به درستی در دیتابیس ذخیره شده است.



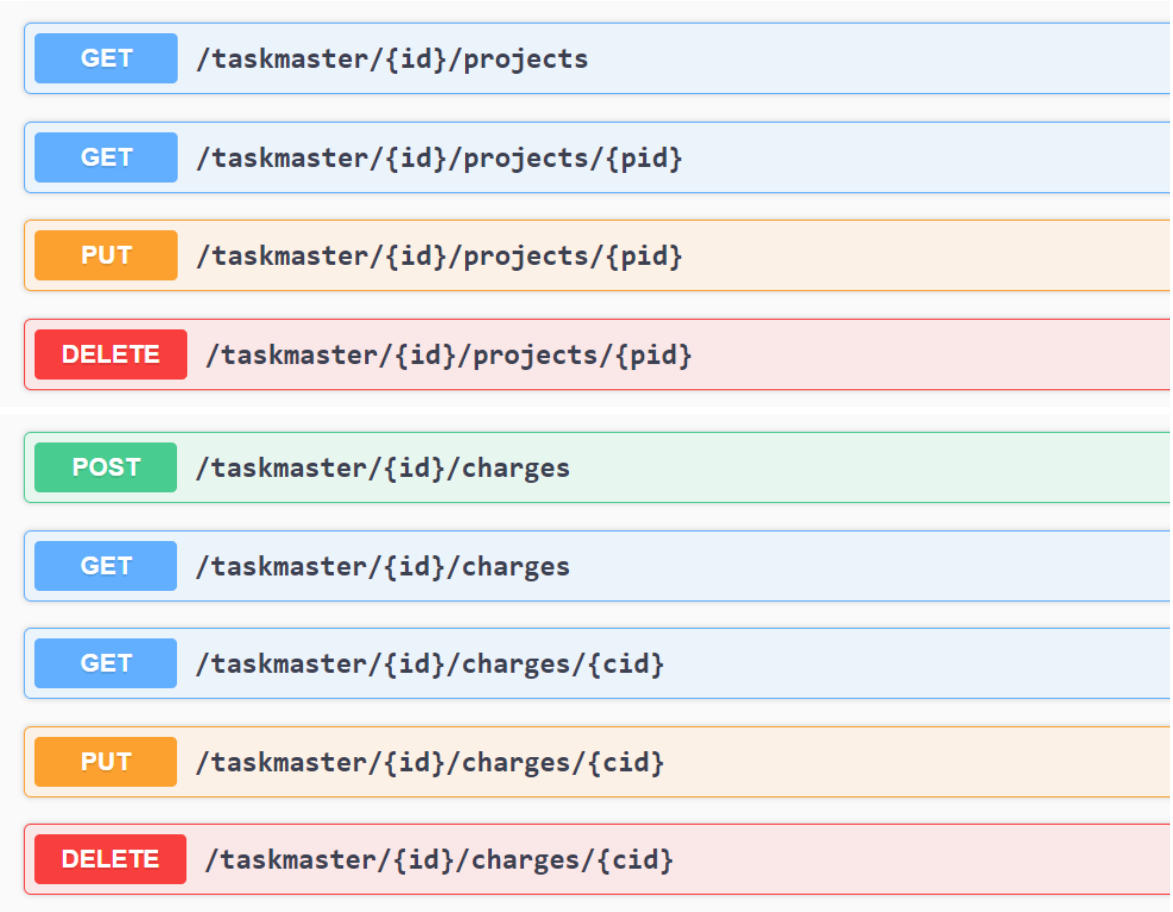

## بخش ششم:

ابتدا اقدام به تعریف موجودیت taskmaster کردم تا بتوان دو قابلیت در نظر گرفته شده برای آن را پیاده‌سازی کرد. (البته قبل از این مورد در پوشه db تمام entity های لازم را تعریف کرده و کلاس‌های dto را نیز طبق طراحی که در قسمت دوم قابل مشاهده می‌باشد پیاده‌سازی کردم). همچنین برای ساده‌تر شدن تست کردن با ابزار Swagger در تعریف فیلدهای entity ها از ApiResponse استفاده کردم تا در موقع وارد کردن اطلاعات به صورت json نام فیلدها به طور اتومات از قبل در باکس مورد نظر وجود داشته باشند.

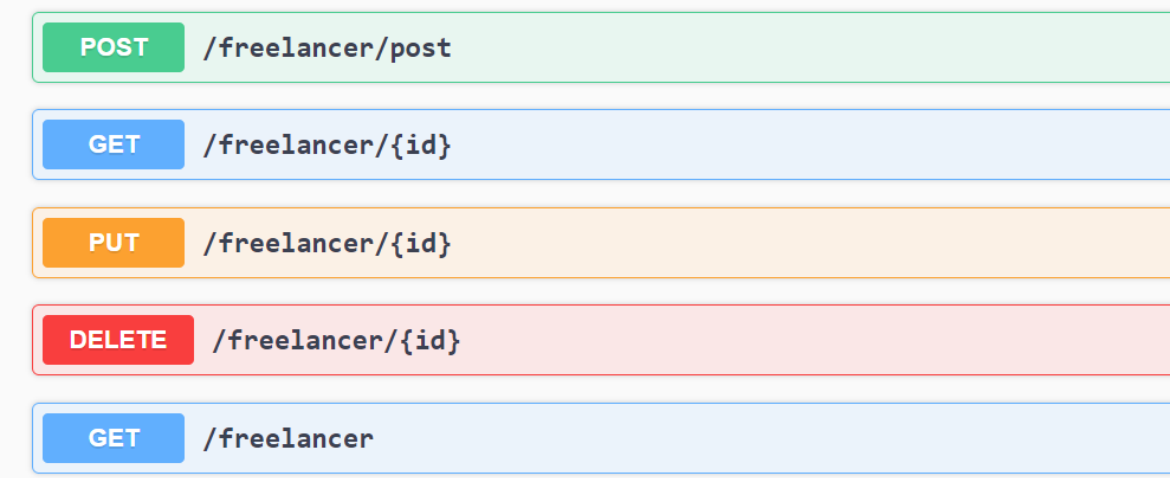
شکل زیر مربوط به خود taskmaster می‌باشد:

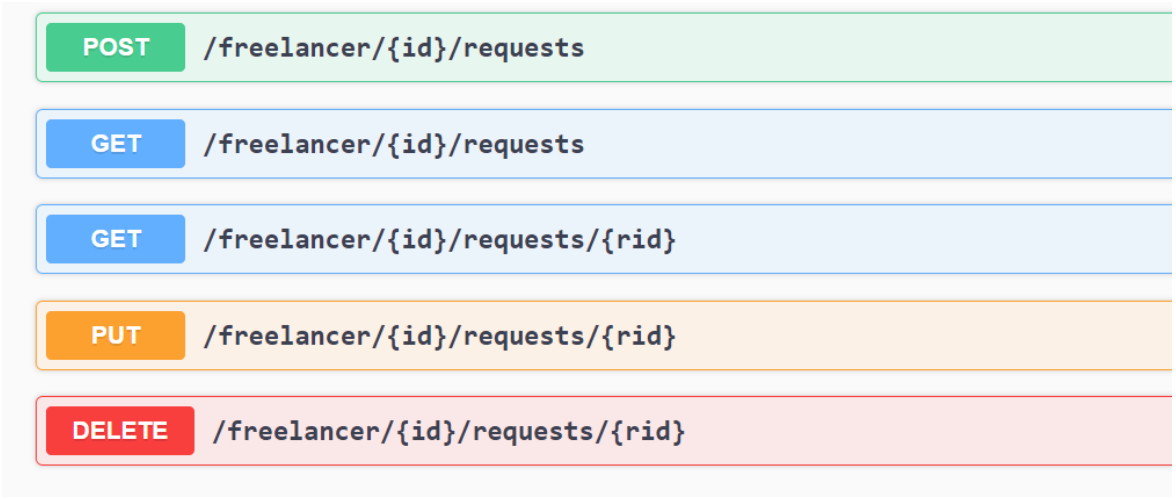


سپس اقدام به تعریف ۴ عمل مورد نیاز برای موجودیت‌های Project و Charge که مربوط به Taskmaster هستند کردم. لازم به ذکر است که برای هر کدام از این موارد علاوه بر متد GetAll که استاد در نظر داشتند متد GetByID نیز در نظر گرفته شده است و به عنوان utility در متدهای update و delete استفاده شده است. نتیجه در شکل زیر قابل مشاهده است:




در گام آخر نیز اقدام به تعریف موجودیت freelancer دقیقا مانند taskmaster کرده و قابلیت request نیز به آن اضافه کردم. لازم به ذکر است که هر درخواست برای یک پروژه می‌باشد لذا در تعریف موجودیت‌های project و request این مورد لحاظ شده است:




POST /freelancer/{id}/requests
GET /freelancer/{id}/requests
GET /freelancer/{id}/requests/{rid}
PUT /freelancer/{id}/requests/{rid}
DELETE /freelancer/{id}/requests/{rid}

## مشکلات و توضیحات تکمیلی

به نظرم اصلی آن بخشی که گفته شده کدها را از روی سایت کپی کرده و مراحل را از آنجا انجام دهیم بشدت دارای باگ‌ها می‌باشد و توصیه می‌شود که مقاله‌ای مناسب‌تر برای قسمت یادگیری کار با nest (قبل از پیاده‌سازی اصلی قابلیت‌ها) در نظر گرفته شود.

## آنچه آموختم

* نحوه استفاده از با dbeaver به منظور کار کردن با پستگرس
* ساختار کلی nest و نحوه کار کردن با آن
* متصل کردن کد بخش بک‌اند به پایگاه‌داده های sqlite و postgres
* روال تبدیل یک نمودار ER به کد سمت بک‌اند و پایگاه‌داده واقعی