

گزارش تمرین اول درس رایانش ابری (آشنایی با برخی از خدمات ابری)

محمدجواد زندیه (۹۸۳۱۰۳۲)

بخش ۱: مراحل **sign up** و **login** برای ساخت اکانت و مدیریت آن







برای مدیریت کاربران استفاده کننده از سامانه، از سرویس **auth0** استفاده شده است.

۱. ابتدا در قسمت Applications باید یک application با تنظیمات مورد نیاز پروژه خود بسازیم:

Applications

+ Create Application

Setup a mobile, web or IoT application to use Auth0 for Authentication. [Learn more](#) →

	Default App Single Page Application	Client ID: 4xj1ScyfHj3kBZnwCz6oaY0mCP2H5avP		
	Juera Application Machine to Machine	Client ID: Eg46gThxEwWdW4Tdrh0T7gv8hoe7Lt8n		







ما application خود را با نام Juera Application ایجاد کردیم و نوع Machine to Machine را برای آن انتخاب کردیم زیرا میخواستیم که از طریق api هایی که توسط auth0 ارائه شده است استفاده کنیم.

۲. در قسمت APIs یک api به نام Juera API ایجاد کرده ایم که روی پورت ۸۰۰۰ از لوکال ما فعال باشد. تایپ آن نیز به صورت custom قرار داده شده است:

APIs

+ Create API

Define APIs that you can consume from your authorized applications.

	Auth0 Management API System API	API Audience: https://dev-pwj3ar40sb22h0a.us.auth0_		
	Juera API Custom API	API Audience: http://127.0.0.1:8000/		

۳. در قسمت Database یک دیتابیس برای نوع ذخیره سازی و ایجاد کاربران ساخته ایم. برای اهراز هویت کاربرات از شیوه Eamil , Password برای ذخیره سازی کاربران استفاده شده است:

Database Connections

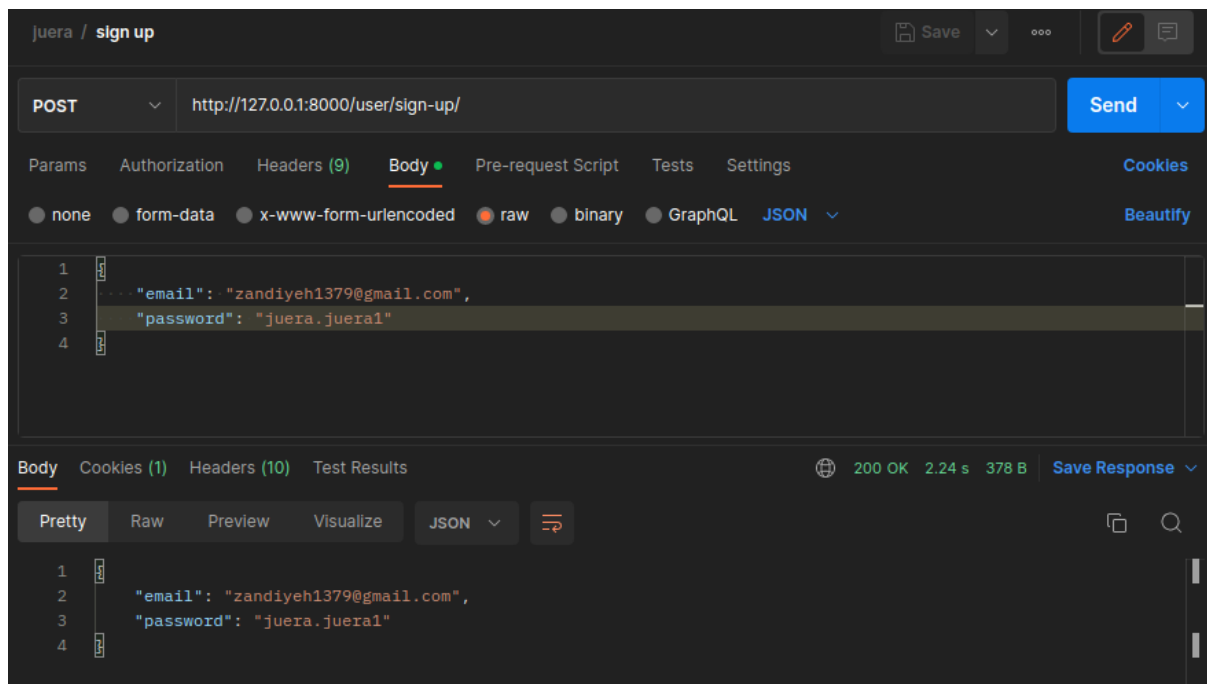
+ Create DB Connection

Securely store and manage username / password credentials either in an Auth0 Database or in your own store. [Learn more](#) →

	Email-Password-Authentication Database	
---	--	---

۴. سپس api های مورد نیاز برای ساخت و ورود به اکانت را در سرویس شماره ۱ خود (service1-api) ایجاد کرده ایم:

مرحله sign up:











مشاهده میکنیم که پس از عملیات sign up یک کاربر در بخش Users پنل کاربری auth0 اضافه میشود که last login آن never است.

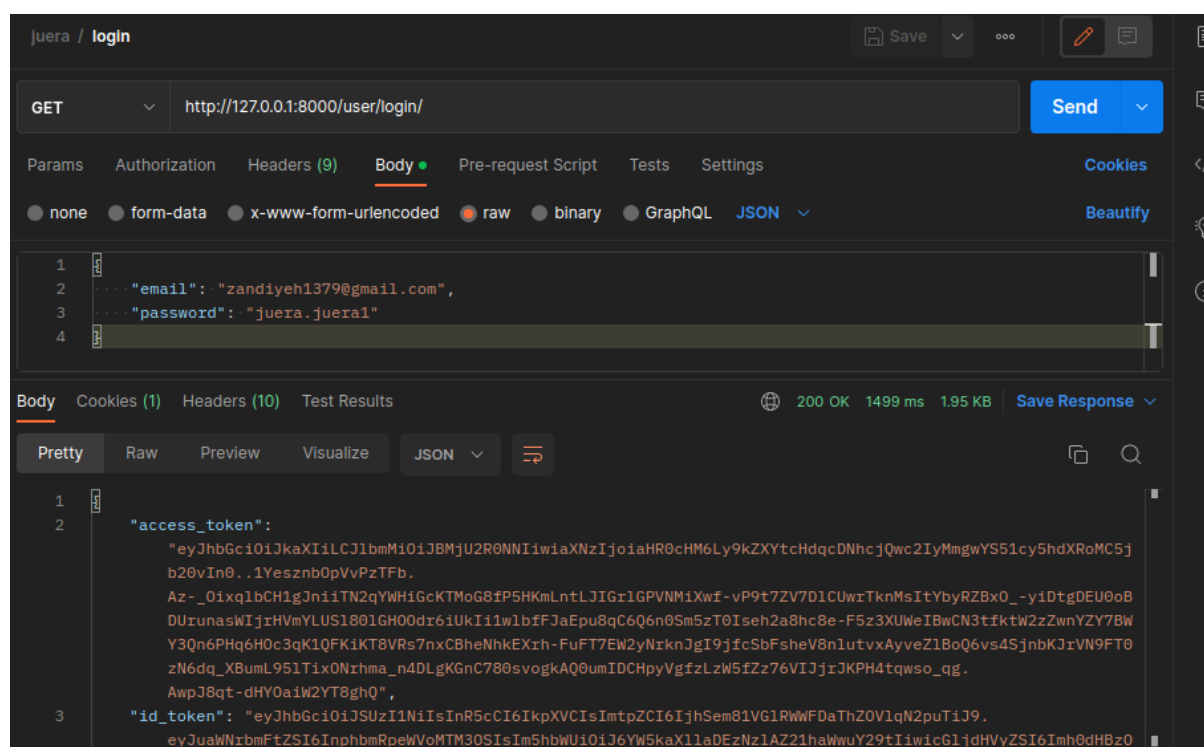
Users

[+ Create User](#)

An easy to use UI to help administrators manage user identities including password resets, creating and provisioning, blocking and deleting users. [Learn more →](#)

Name	Connection	Logins	Latest Login	
 amir@gmail.com amir@gmail.com	Email-Password-Authentication	5	2 days ago	
 fasfa@fsadf.casdgfdsgfscom fasfa@fsadf.casdgfdsgfscom	Email-Password-Authentication	1	10 days ago	
 dfasfasdfas@dsffads.cdsafdasf dfasfasdfas@dsffads.cdsafdasf	Email-Password-Authentication	1	10 days ago	
 zandiyeh1379@gmail.com zandiyeh1379@gmail.com	Email-Password-Authentication	0	never	

مرحله :login



در این مرحله پس از انجام عملیات login یک access_token به همراه id_token و مابقی اطلاعات بازگردانده میشود که برای استفاده از api هایی که نیاز به login دارند، از این access_token در هدر ریکوئست‌ها استفاده میکنیم. این access_token حاوی اطلاعات کاربری است که لاگین کرده است. تعداد لاگین‌های کاربران نیز در پنل auth0 نمایش داده می‌شود.

Users

[+ Create User](#)

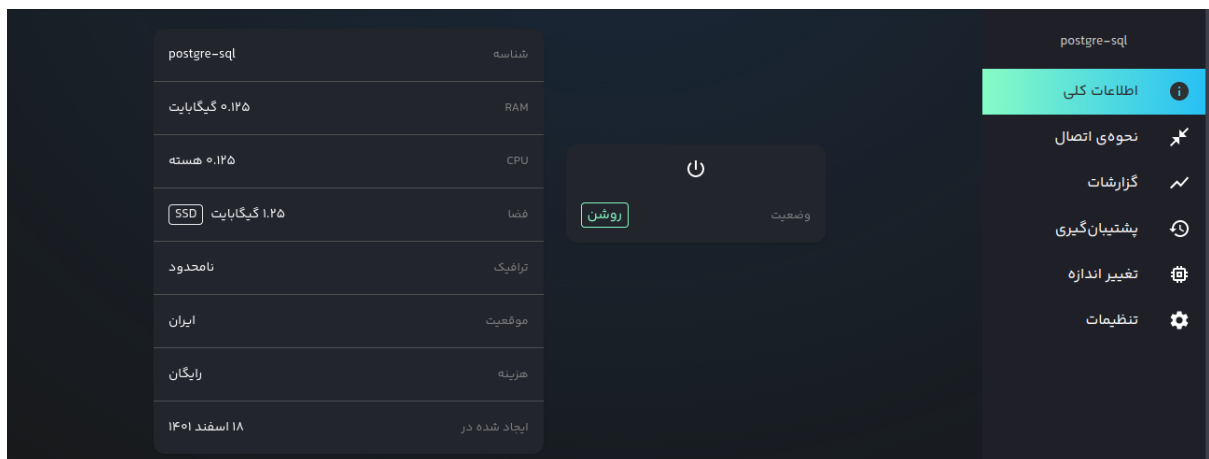
An easy to use UI to help administrators manage user identities including password resets, creating and provisioning, blocking and deleting users. [Learn more →](#)

Name	Connection	Logins	Latest Login	
<div>ZA</div> <div>zandiyeh1379@gmail.com</div> <div>zandiyeh1379@gmail.com</div>	Email-Password-Authentication	1	3 minutes ago	...
<div>AM</div> <div>amir@gmail.com</div> <div>amir@gmail.com</div>	Email-Password-Authentication	5	2 days ago	...
<div>FA</div> <div>fasfa@fsadf.casdgfdsgfscom</div> <div>fasfa@fsadf.casdgfdsgfscom</div>	Email-Password-Authentication	1	10 days ago	...
<div>DF</div> <div>dfasfasdfas@dsffads.cdsafdasf</div> <div>dfasfasdfas@dsffads.cdsafdasf</div>	Email-Password-Authentication	1	10 days ago	...

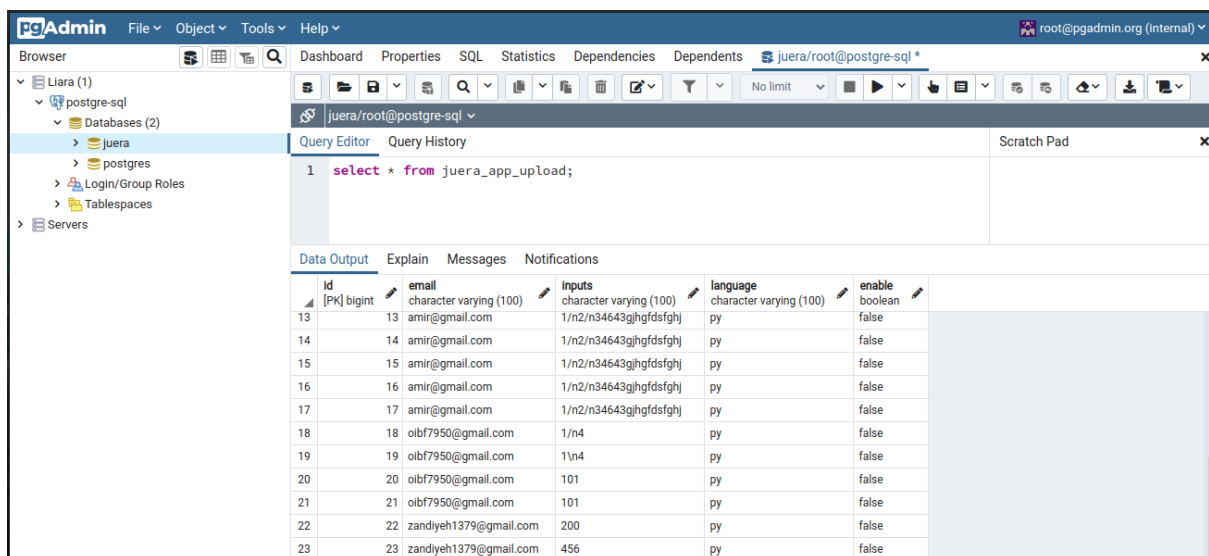
بخش ۲: مرحله آپلود فایل

در سرویس شماره ۱ یک api فراهم شده است که از طریق آن میتوان یک فایل را به همراه زبان برنامه نویسی و ورودی‌های آن نگهداری کرد. برای نگهداری اطلاعات آپلود نیاز به یک پایگاه داده داریم. همچنین برای نگهداری فایل‌های آپلود شده نیاز به یک محل ذخیره سازی فایل داریم.

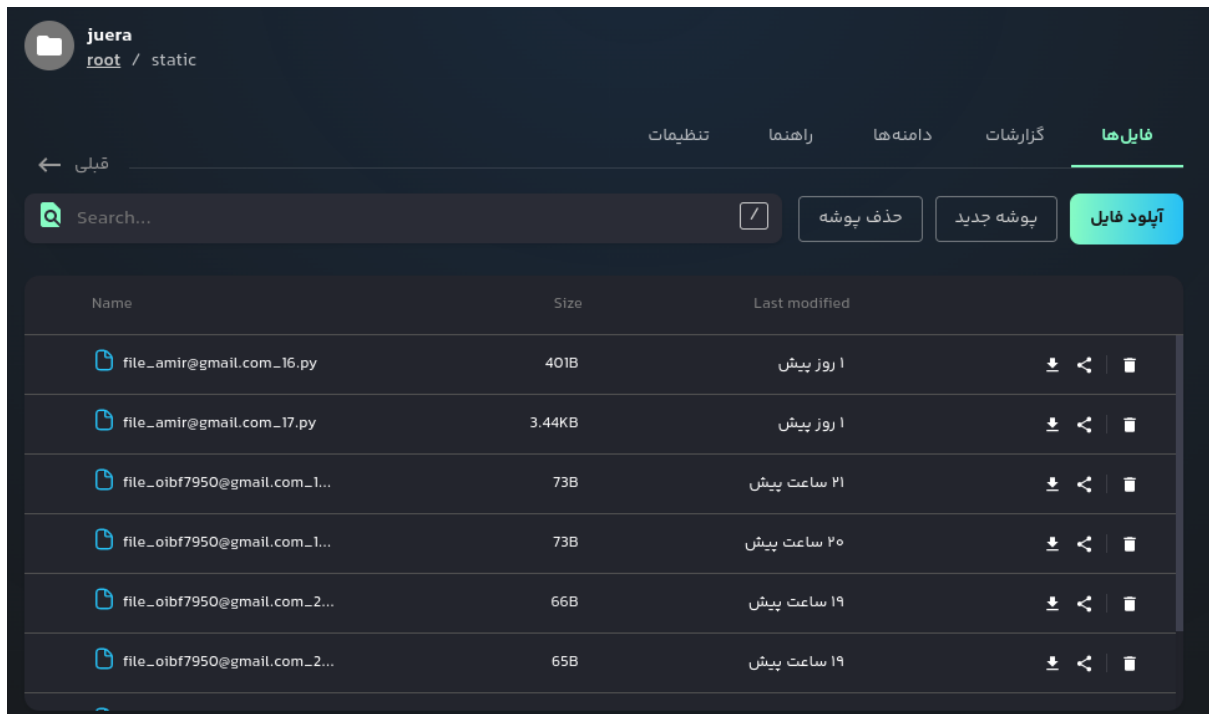
۱. ذخیره در پایگاه داده: برای file storage ما از سرویس لیارا استفاده کرده ایم. ابتدا در پنل کاربری در قسمت دیتابیس یک پایگاه داده PostgreSQL ایجاد میکنیم. سپس یک پایگاه داده به نام juera ایجاد میکنیم و اطلاعات خود را در آن ذخیره میکنیم.



برآ مشاهده جداولی که توسط سرویس ۱ در پایگاه داده ایجاد کرده ایم هم میتوان از قسمت "نحوه اتصال" راه‌اندازی PGAdmin را فعال میکنیم و میتونیم در PGAdmin جداول و رکوردهای ایجاد شده را مشاهده کرد. توجه شود که تمام جداولی که ایجاد میکنیم دارای پیشوندی به نام juera_app خواهند بود.

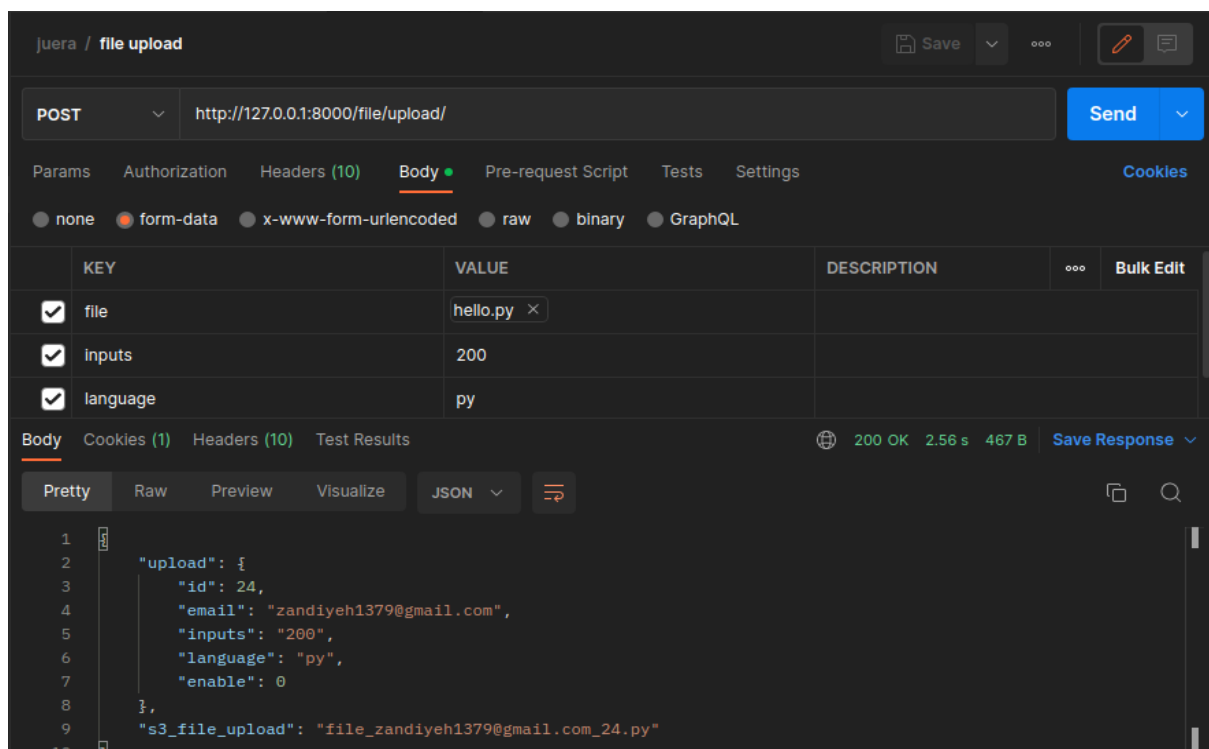


۲. ذخیره سازی فایل: برای ذخیره سازی فایل نیز از سرویس لیارا استفاده میکنیم. در قسمت ذخیره سازی ابری یک bucket به نام juera ایجاد کرده ایم که محل ذخیره سازی فایل های آپلود شده میباشد. فایل های آپلود شده را در قسمت static نگهداری میکنیم.



برای نام فایل های ذخیره شده هم از الگوی زیر استفاده شده است:

`file_{user_email}_{upload_id}.{upload_language}`

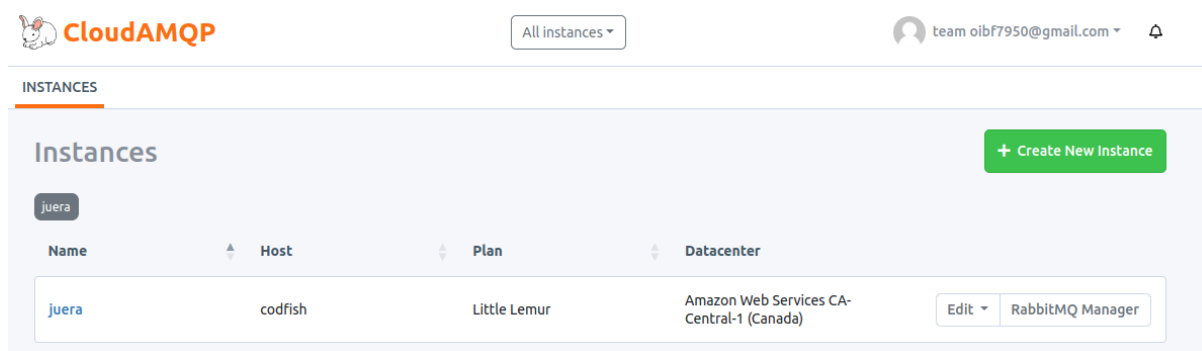


بخش ۳: اجرای فایل

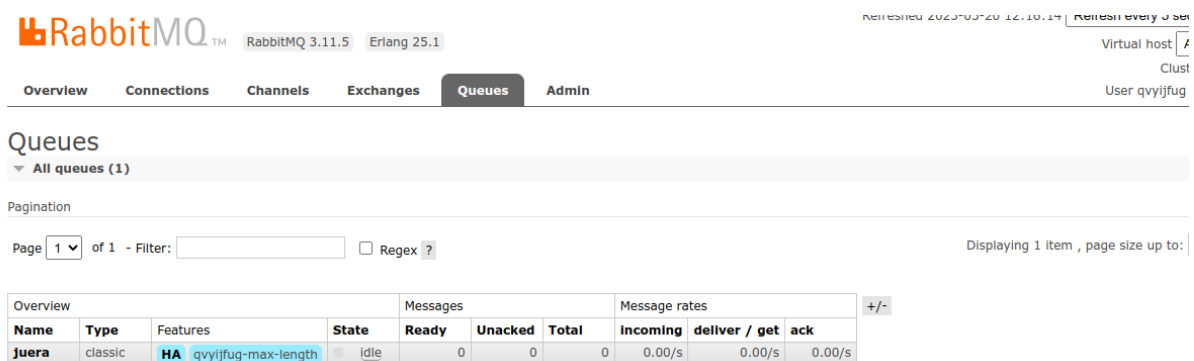
برای اجرای فایل یک ریکوئست به api ایجاد شده در سرویس ۱ زده میشود که در آن آیدی فایل آپلود شده قرار داده میشود و اگر فایل متعلق به آن کاربر باشد و enable آن باشد اجازه اجرای آن را دارد. اجرای فایل در مراحل زیر صورت میگیرد:

۱. ذخیره اسم فایل درخواست شده برای اجرا در یک Message Broker به نام **RabbitMQ**:

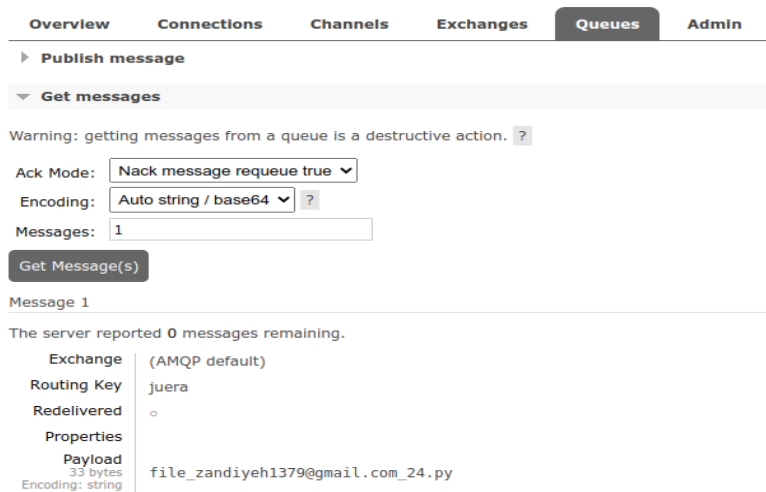
در RabbitMQ یک صف به نام juera ایجاد کرده ایم و فایل هایی که باید اجرا شوند را از طبق شناسه نام آنها در این صف قرار میدهیم.

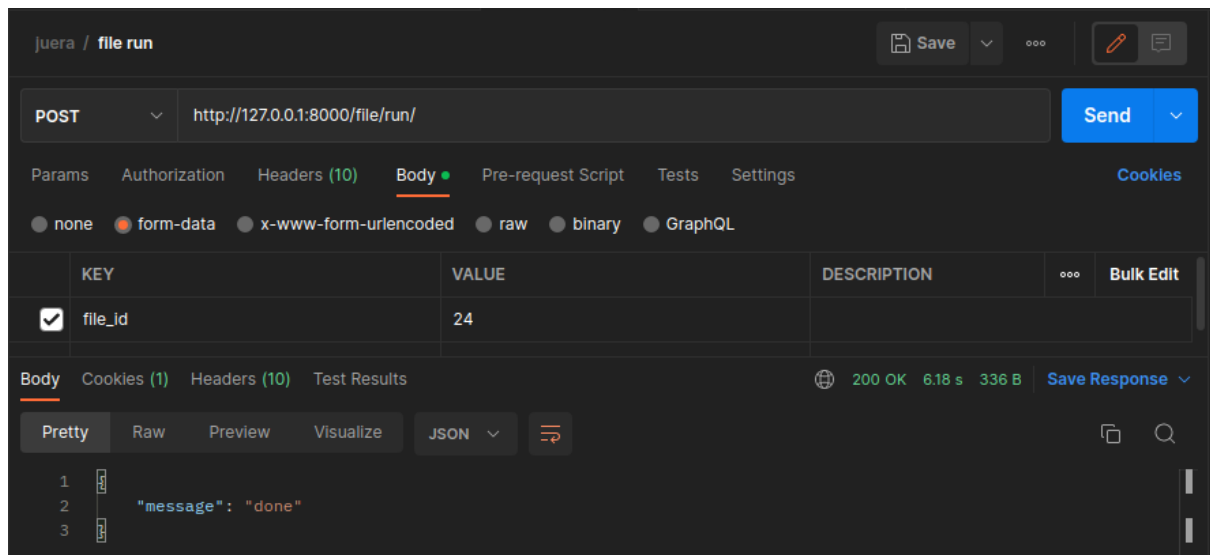


با استفاده از RabbitMQ Manager میتوان وضعیت صف مورد نظر را مشاهده کرد:



همچنین میتوان رکوردهای داخل صف را نیز مشاهده کرد (اگر رکوردی از صف برداشته و اجرا شود دیگر قابل مشاهده نیست).





۲. برداشت شناسه فایل از صف و ساخت job:

برای برداشتن و ساخت job سرویس شماره ۲ (service2-job-creator) را ایجاد کرده ایم. در این سرویس ابتدا به پایگاه داده ای که پیش از این گفته شده بود وصل میشویم و پس از آن تمام شناسه‌های داخل صف را برمیداریم و از روی آنها فایل مربوطه را از file storage استخراج میکنیم. سپس با ترکیب زبان، ورودی ها و محتوای فایل یک query string ایجاد میکنیم و آن را تحت عنوان یک رکورد در پایگاه داده و جدول job قرار میدهیم.

```
javad@javad-HP-350-61:~/Desktop/Files/cloud/hw/hw1/juera/service2-job-creator$ python3 job_creator.py
log*: waiting for messages. to exit press ctrl+c
log: job created for upload {"email": "zandiyeh1379@gmail.com", "file_id": "24", "language": "py"}
```

1	select * from juera_app_job;	
---	------------------------------	--

Data Output	Explain	Messages	Notifications
id	job		
[PK] bigint	character varying (10000)		
22	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=200		
23	code=a%2C+b+%3D+int%28input%28%29%29%2C+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Ba%2Bb%7D+times+to+you%22%29&language=py&i		
24	code=a%2C+b+%3D+int%28input%28%29%29%2C+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Ba%2Bb%7D+times+to+you%22%29&language=py&i		
25	code=a%2C+b+%3D+int%28input%28%29%29%2C+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Ba%2Bb%7D+times+to+you%22%29&language=py&i		
26	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=101		
27	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=101		
28	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=101		
29	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=200		
30	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=456		
31	code=number+3D+int%28input%28%29%29%0A%0Aprint%28f%22hello+for+%7Bnumber%7D+times+to+you%22%29%0A&language=py&input=200		

صف هم متناسب با این فرایند خالی از شناسه گفته شده میشود:

▼ Get messages

Warning: getting messages from a queue is a destructive action

Ack Mode:

Nack message requeue true

Encoding:

Auto string / base64

Messages:

1

Get Message(s)

Queue is empty

Close

بخش ۴: اجرای فایل

در سرویس شماره ۳ (service3-job-runner) به صورت بازه‌های ۳ ثانیه ای تمام job هایی که در وضعیت none هستند از پایگاه داده خوانده میشود و با ارسال job به سرویس [codex](#) خروجی مورد نظر برای کاربر ایمیل میشود. در این میان result هم با شروع فرایند اجرا میشود در پایگاه داده و با پایان فرایند و گرفتن پاسخ از codex به وضعیت done تغییر وضعیت میدهد.

۱. سرویس codex: برای استفاده از این سرویس نیاز به ساخت اکانت نیست و صرفا با ریکوئست زدن به <https://api.codex.jaagrav.in> میتوان خروجی درخواست خود را دید. دقت شود که فرمت دیتای ارسالی باید به صورت دیکشنری از code, language, input باشد.

Query Editor Query History

```
1 select * from juera_app_result;
```

Data Output Explain Messages Notifications

	id [PK] bigint	output character varying (10000)	status character varying (100)	executed_date timestamp with time zone	job_id bigint
65	74	hello for 456 times to you	done	2023-03-19 23:59:05.649605+00	34
66	75	hello for 456 times to you	done	2023-03-19 23:59:13.809048+00	34
67	76	hello for 456 times to you	done	2023-03-20 00:04:13.740352+00	34
68	77	hello for 456 times to you	done	2023-03-20 00:10:31.581761+00	34
69	78	hello for 456 times to you	done	2023-03-20 00:23:50.091994+00	34
70	79	hello for 456 times to you	done	2023-03-20 00:24:00.683383+00	34
71	80	hello for 456 times to you	done	2023-03-20 00:24:07.026928+00	34
72	81	error for connecting to Codex	done	2023-03-20 00:24:18.203125+00	34
73	82	hello for 456 times to you	done	2023-03-20 12:51:08.413901+00	34
74	83	hello for 200 times to you	done	2023-03-20 12:51:28.241951+00	35
75	84	hello for 200 times to you	done	2023-03-20 12:51:35.155087+00	36

۲. سرویس [mailgun](#): برای ارسال ایمیل نتیجه فرایند انجام شده از این سرویس استفاده میشود. در قسمت Overview باید آدرس ایمیل هایی که قرار است به آنها ایمیل زده شود، در قسمت Recipient وریفای شوند، در غیر این صورت ایمیل برای آنها ارسال نخواهد شد.



بخش ۵: گرفتن نتایج

یک api در سرویس اول فراهم شده است که نتایج اجراهای انجام شده برای کاربر را به او برمیگرداند:

