

دانشگاه صنعتي امیرکبیر  
(پلی تکنیک تهران)

دانشكده مهندسی کامپیوتر

گزارش تمرین اول درس رایانش ابری

نگارش

بردیا اردکانیان

استاد

دکتر سید احمد جوادی

آبان و ۱۴۰۱

شرح راهبرد پیاده‌سازی

# طرح پروژه

در پیاده‌سازی پروژه از چاچوب جنگو استفاده شده است. علت این انتخاب سادگی، در دسترس بودن و سازگاری زبان پایتون با نیازمندی‌های پروژه می‌باشد.

مابقی طرح پروژه و سرویس‌های استفاده شده به شرح ذیل می‌باشد.

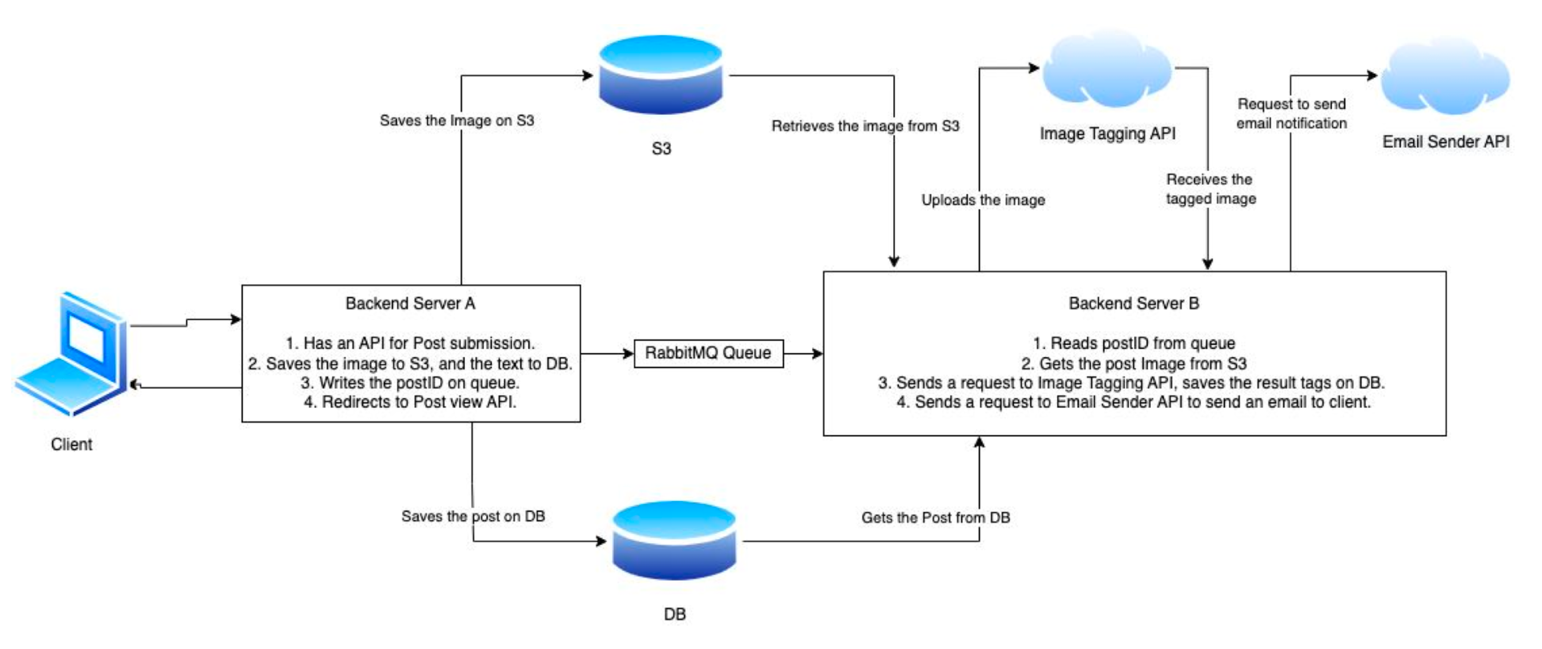
* Cloud Hosting Service: ArvanCloud
* Database as a Service: MongoDB
* Simple Storage Service: ArvanCloud S3 Storage
* Message Broker Service: RabbitMQ
* Image Classification Service: Imagga
* Email Delivery Service: Mailgun

همچنین در استقرار پروژه از Docker و Kubernetes استفاده‌شده است.

به منظور راه اندازی پروژه می‌بایست ماژول‌های ذیل دانلود شده باشند.

* boto3==1.24.28
* botocore==1.27.28
* Django==4.1.3
* pika==1.3.1
* pymongo==4.3.2
* requests==2.28.1

معماری پروژه تماما با معماری پیشنهاد داده شده مطابقت دارد.



عکس ۱

پیاده‌سازی پروژه

# سرویس اول – ثبت و استعلام آگهی

به منظور پیاده‌سازی API به منظور ثبت آگهی ابتدا می‌بایست تمامی عملیات ساختن، خواندن، به‌روز کردن و پاک کردن را برای پایگاه‌داده پیاده سازی کرد. به همین دلیل در فایل utils.py توابعی به منظور ایجاد ارتباط و دستکاری داده ایجاد شدند.

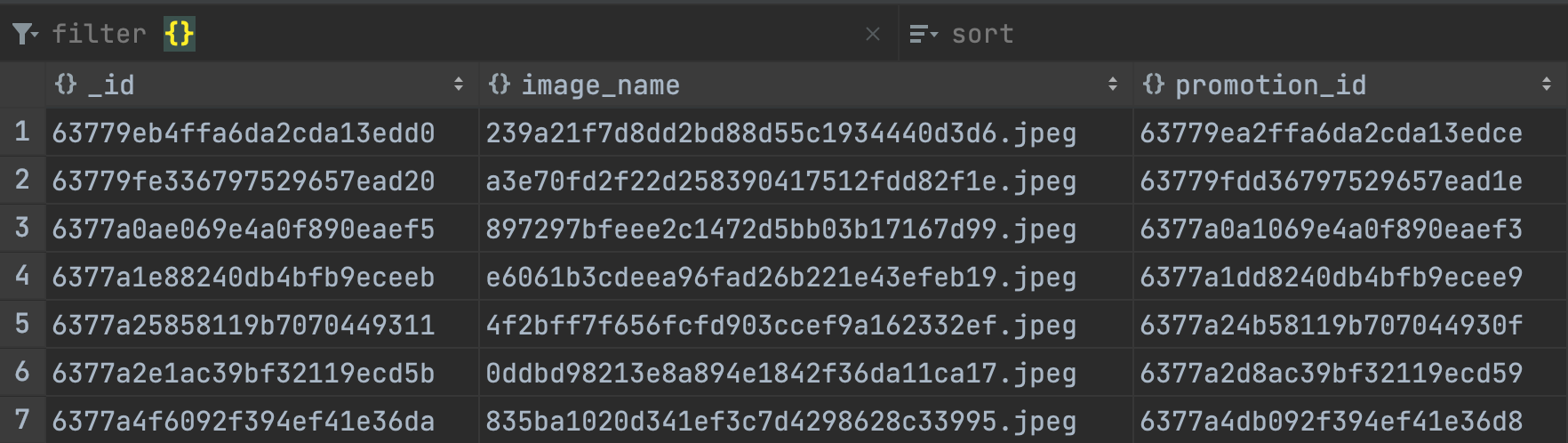
این توابع در مرحله اول با کمک Connection String که از سایت [cloud.mongodb.com](http://cloud.mongodb.com/) دریافت کرده‌ایم با پایگاه داده ارتباط برقرار می‌کنیم. این ارتباط توسط پروتکل TCP/IP ایجاد می‌شود. در ادامه دو Collection به نام‌های Promotion و Image ایجاد می‌کنیم. شرح هر یک مختصرا به شکل زیر می‌باشد.

مجموعه آگهی «Promotion»: این مجموعه به منظور نگه‌داری آگهی‌های ثبت شده می‌باشد. به این شکل که برای هر آگهی اطلاعات زیر را ثبت می‌کند.

* \_id
* Description
* Email
* State
* Category

داده \_id به صورت اتوماتیک تولید شده و یک کلید منحصر به فرد متشکل از کاراکتر‌های انگلیسی و اعداد را ایجاد می‌کند و به هر شئ نسبت می‌دهد. اطلاعات توضیحات و ایمیل در ابتدا تولید شدن شئ پر می‌شوند و وضعیت شئ به حالت PROCESSING تعریف می‌شود. همچنین داده دسته‌بندی تا موقعی که دسته‌بندی توسط سرویس دیگر صورت گیرد به شکل NULL می‌ماند.

مجموعه تصویر «Image»: هر اگهی می‌تواند یک یا چند تصویر داشته باشد و مجموعه تصویر نام عکس ذخیره شده در مخزن ابرآروان را به کلید یکتا آگهی‌ها می‌نگارد. این نگاشت به منظور بازیابی تصاویر هر آگهی الزامی می‌باشد و باعث می‌شوند پروژه Scalable باشد چرا که در آینده می‌توان نام فایل‌های دیگری را نیز به هر آگهی نگاشت.



عکس ۱-۱

بعد از فرستاده شدن درخواست ایجاد آگهی ابتدا مجموعه آگهی تولید می‌شود و بعد با کمک توابع پیاده سازی شده با کتابخانه boto و ارتباط گیری با مخزن ابری ابرآروان تصویر فرستاده شده در درخواست بارگیری می‌شود، به صورت محلی ذخیره می‌شود و در نهایت در مخزن ابری بارگزاری می‌شود. در این فرایند نام تصویر بارگزاری شده به صورت تصادفی و به اندازه ۱۶ کاراکتر تولید می‌شود. در نهایت نام تولید شده به کلید یکتا آگهی ثبت شده نگاشت می‌شود و در پاسخ کلید یکتا سرویس برگردانده می‌شود. مثالی از خروجی برنامه:

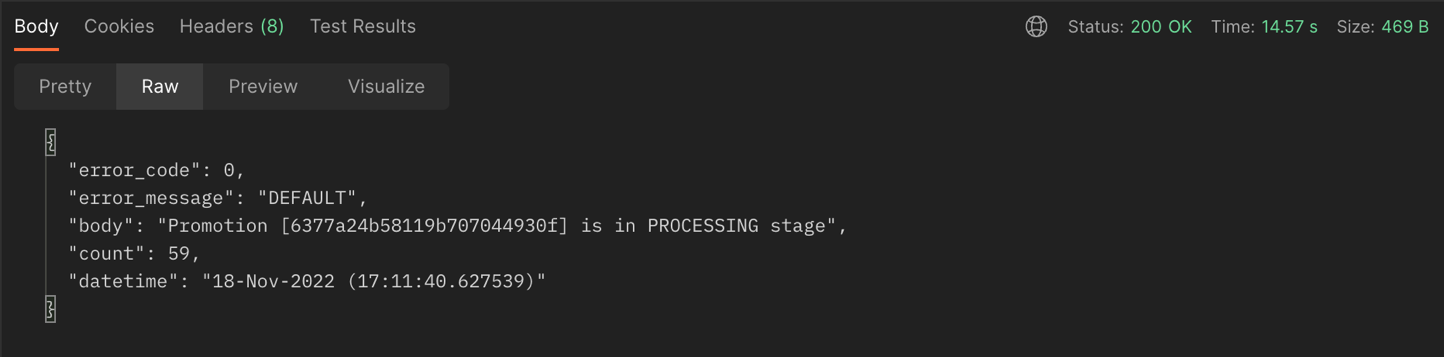


عکس ۱-۲

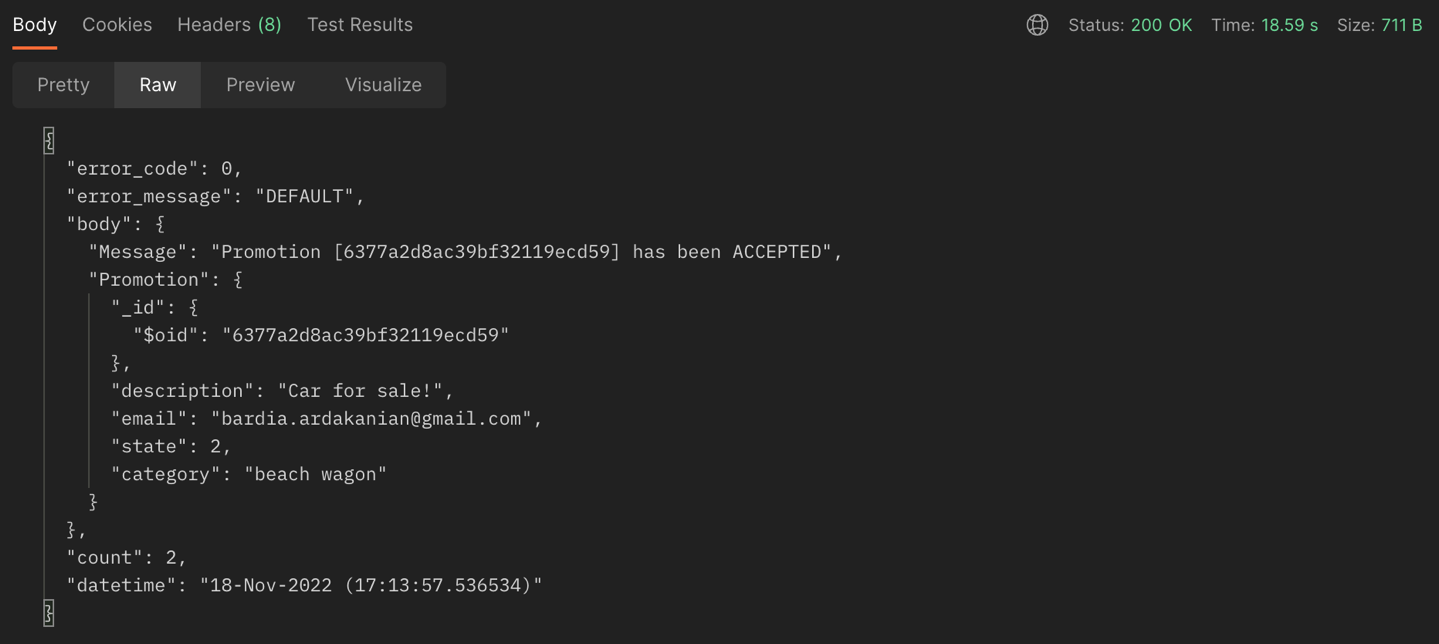
در صورتی که ثبت آگهی با مشکل رو به رو شود خطایی نظیر علت رخداد مشکل چاپ می‌شود.

بعد از ثبت شدن آگهی ارتباطی با صف RabbitMQ برقرار می‌شود و شناسه یکتا آگهی در صف مورد نظر ثبت می‌شود. برای پیاده سازی این بخش ابتدا توابعی به منظور برقراری ارتباط با RabbitMQ ایجاد شدند و با استفاده از API مستند‌سازی شده در [سایت RabbitMQ](https://stingray.rmq.cloudamqp.com/#/) صفی ایجاد می‌شود. در نهایت شناسه یکتا آگهی در صف قرار داده می‌شود.

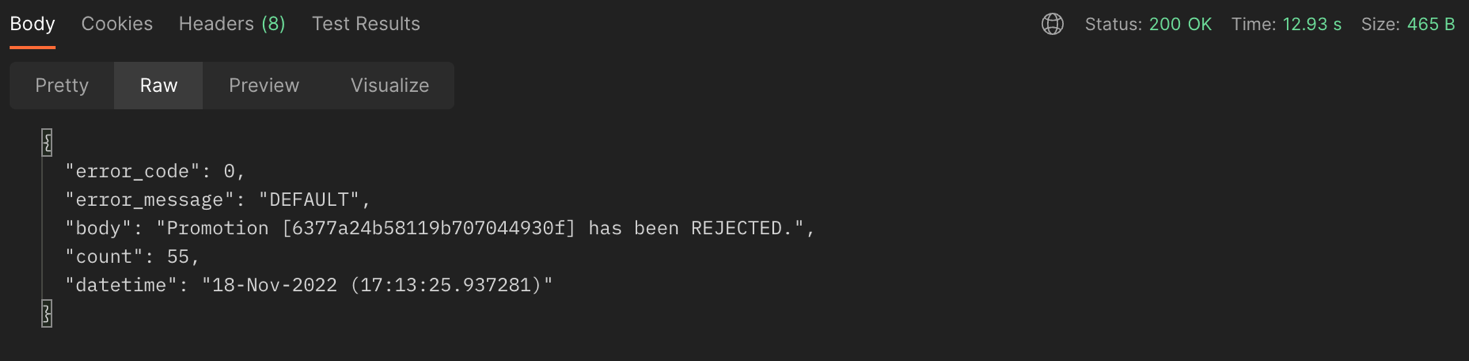
سرویس استعلام شناسه یکتا آگهی را دریافت می‌کند و یکی از خروجی‌های زیر را چاپ می‌کند.



عکس ۱-۳



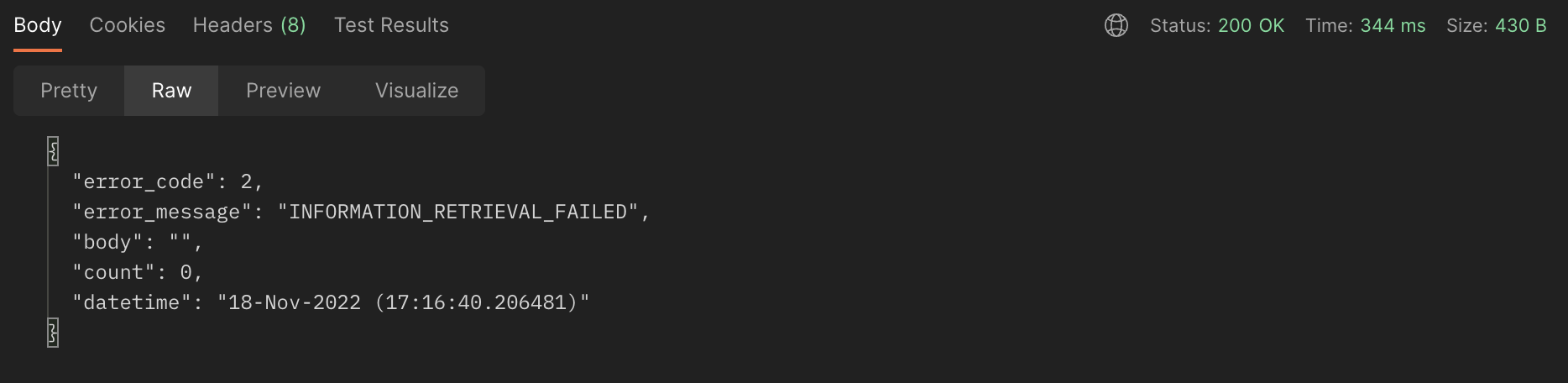
عکس ۱-۴



عکس ۱-۵

همانطور که مشاهده می‌کنید در صورت قبول شدن آگهی برچسب آن نیز برگردانده می‌شود. پیاده‌سازی این بخش با جزئیات توضیح داده خواهد شد. همچنین در صورت تغییر وضعیت آگهی به ایمیل ثبت شده در پایگاه داده اطلاع رسانی خواهد شد.

در ادامه در صورتی که استعلام با مشکل رو به رو شود خطایی متناظر با علت خطا برگردانده می‌شود. به عنوان مثال خطای زیر برگردانده می‌شود.



عکس ۱-۶

# سرویس دوم – دسته‌بندی تصویر و اطلاع رسانی از طریق ایمیل

سرویس دوم ابتدا با ایجاد اتصال با صف RabbitMQ تک به تک شناسه‌های یکتا آگهی‌ها را دریافت می‌کند. بعد از دریافت شناسه یکتا و استعلام نام فایل ذخیره شده در مخزن ابری ابرآروان، آن را بارگیری می‌کند و در پوشه‌ای ذخیره می‌کند. بعد از ذخیره کردن فایل با فراخانی سرویس Imagga و فرستادن عکس به سرویس ابری مربوطه دسته‌بندی‌های پیشبینی شده را با درصد اطمینان ثبت شده دریافت می‌کند. در نهایت با پیشمایش بر دسته‌بندی‌ها در صورتی که قوانین درج شده براورده شوند دسته بندی‌ که بیشترین درصد اطمینان را دارد به عنوان دسته‌بندی آگهی ثبت می‌شوند و وضعیت آگهی به وضعیت قبول شده تغییر داده می‌شود. در صورتی که قوانین نام برده برآورده نشوند وضعیت آگهی به رد شده تغییر داده می‌شود.

کد پیاده سازی شده.



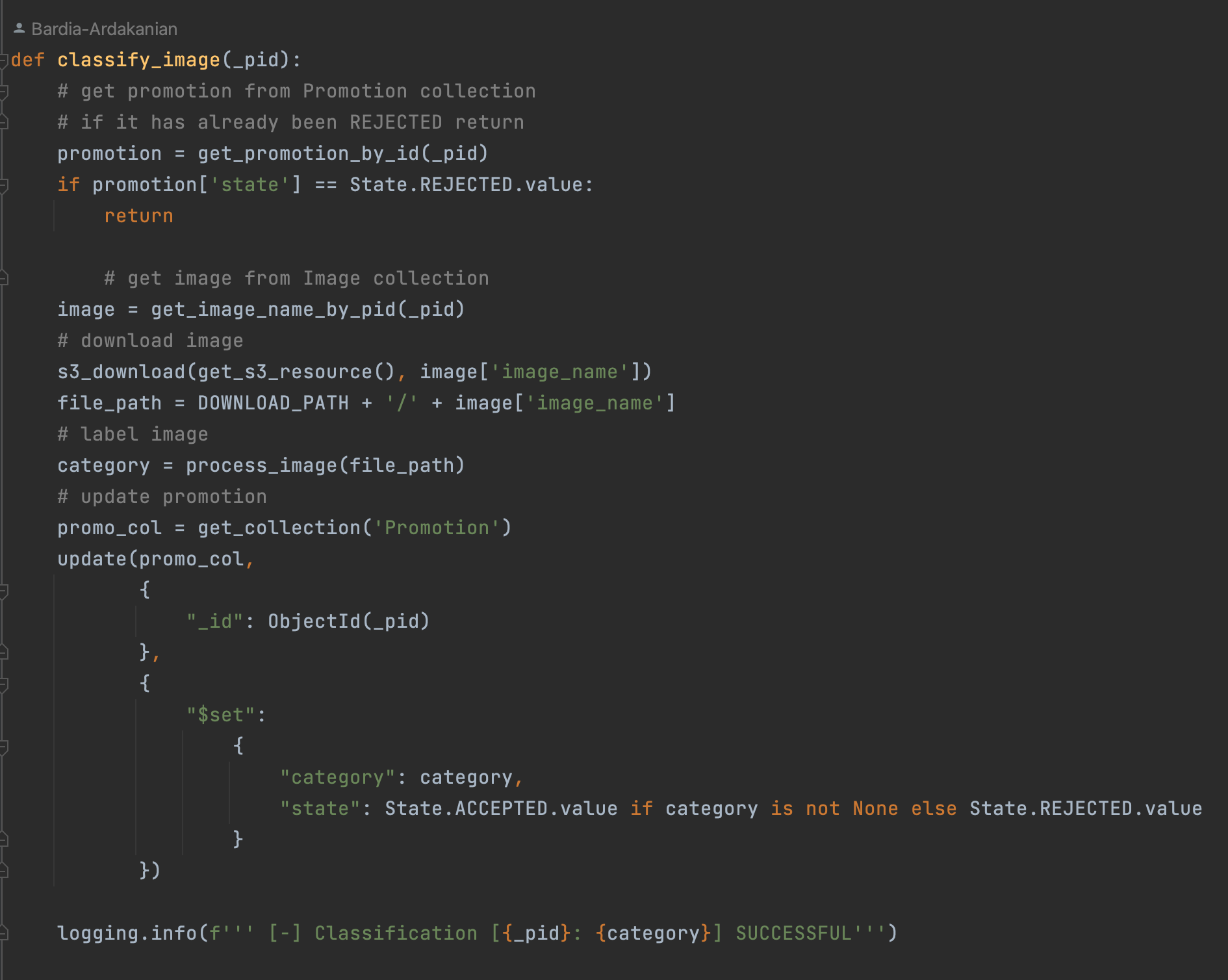
عکس ۱-۷

زمانی که وضعیت آگهی تغییر می‌کند با فرستادن ایمیلی به سازنده آگهی ثبت شده در پایگاه‌داده وضعیت جدید آگهی را اطلاع رسانی می‌کند. طریقه ارسال ایمیل به این صورت است که ابتدا با کمک مشخص کردن دامنه پروژه، کلید خصوصی و گیرنده ایمیل درخواستی را به سرور‌های Mailgun ارسال می‌کند. بعد از ارسال شدن درخواست به سرور مربوطه و پردازش درخواست، ایمیل به گیرنده‌ها ارسال می‌شود. برای اینکه ایمیل در پوشه spam ذخیره نشود می‌بایست نام فرستنده را به no-reply تغییر می‌دادیم.

# چالش‌های پیاده‌سازی

زبان برنامه نویسی:‌ در ابتدا تلاش کردم تا با زمان Go پروژه را پیاده سازی کنم ولی متاسفانه به علت مشکلات اینترنتی و تحریم ایران توسط گوگل موفق به دانلود کردن کتاب‌خانه‌های مورد نظر نشدم و در نهایت با زبان پایتون به پیاده سازی پرداختم.

میزبان ابری: پیاده‌سازی پروژه بر میزبان ابری با چالش‌های بسیاری همراه بود. سرویس ارسال ایمیل در بیشتر ساعات فیلتر می‌باشد و تنها یکبار موفق شدم در میزبان ابری ابرآروان سرویس مربوطه را آزمایش کنم. همچنین استقرار پروژه با کمک Docker و Kubernetes دشوار بود چرا که اولین بار بود با این ابزار‌ها کار می‌کردم و نیاز به آموزش‌های بسیاری داشتم.



عکس ۱-۸

پایگاه داده به عنوان سرویس: ابتدا تلاش کردم تا از سرویس aiven استفاده کنم ولی با توجه به فیلتر بودن سایت موفق به برقرار ارتباط با این سرویس نشدم. در نهایت از MongoDB استفاده کردم که سرعت و عملکرد بهتری نسبت به انتظاراتم داشت. همچنین به علت اینکه یک پایگاه‌داده بدون ساختار بود اضافه و کم کردن اطلاعات به پایگاه داده بسیار آسان بود و تجربه قشنگی را به رقم زد.

مخزن به عنوان سرویس: کار کردن با مخزن ابرآروان با توجه به ضعیف بودن مستندات این سرویس با دشواری‌های بسیاری همراه بود. در نهایت متوجه شدم که مخزن ابری ابرآروان مانند مخزن ابری آمازون سرویس می‌دهد و تواستم با مستند‌سازی‌های آمازون به پیاده‌سازی بخشی از پروژه بپردازم. کتابخانه استفاده شده نیز مخصوص مخزن ابری آمازون می‌باشد.

در پیاده سازی‌های اولیه متوجه شدم فایل‌ها به صورت خراب و مشکل‌دار بارگزاری می‌شدند که دشواری‌های بسیاری را ایجاد کرد. در انتها با مشخص کردن تعدادی پرچم در API کتابخانه boto موفق شدم این مشکل را حل کنم.

ارسال ایمیل به عنوان سرویس: متاسفانه این سرویس بدون فیلترشکن بلا استفاده بود و همین امر سبب شد تا تست کردن برنامه با مشکلاتی رو به رو شود. چرا که با اتصال فیلترشکن امکان استفاده از RabbitMQ از برنامه سلب می‌شد. علت این امر firewall سرویس RabbitMQ می‌باشد و جلوی استفاده از فیلترشکن را می‌گیرد. به همین علت استقرار پروژه در میزبان ابری بسیار چالش برآنگیز شد چرا که فعال کردن فیلترشکن در میزبان ابری مشکل است و دو سرویس ذکر شده در استفاده از فیلترشکن با هم تناقض دارند.

# نتیجه نهایی

تمامی بخش‌های پروژه با موفقیت پیاده‌سازی و تست شده‌اند ولی با توجه به فیلتر بودن بعضی سرویس‌ها استقرار پروژه در میزبان ابری مشکل است. به هر حال پیاده‌سازی کامل می‌باشد ولی امکان تست برنامه در میزبان ابری ممکن نمی‌باشد.