" به نام خدا "

گزارش پروژه اول رایانش ابری (آشنایی با برخی از خدمات ابری)

[مریم گلی - شماره دانشجویی: ۹۸۳۱۰۵۴]

همان طور که در دستور کار ذکر شده است، این پروژه از دو سرویس اصلی تشکیل شده که کد مربوط به آن ها در فایل های Service-1.py و Service-2.py قرار داده شده است. همچنین تعدادی فایل دیگر برای مدیریت قسمت های مختلف و سرویس های ابری مورد استفاده، ایجاد شده که در ادامه به توضیح آن ها می پردازیم.

- فایل های جانبی

نام فایل	توضیحات
DB_handler.py	هدف: مدیریت و ارتباط با پایگاه داده برای ایجاد پایگاه داده از سرویس لیارا به عنوان DBaaS استفاده شده است که پایگاه داده ی MariaDB را پشتیبانی می کند. در این پایگاه داده یک جدول به نام advertisement وجود دارد که دارای ۵ ستون با نام های id، description، email، state و category است. id یک شناسه ی منحصر به فرد (primary key) برای آگهی ثبت شده است که به صورت auto-increment در دیتابیس ثبت می شود. همچنین ستون های email و description توسط ورودی هایی که کاربر در صفحه ی مربوط به ثبت آگهی جدید وارد می کند، پر می شود. ستون state، سه مقدار "accepted" ، "failed" و "pending" را می تواند اختیار کند که هر کدام نشان دهنده ی state فعلی آگهی هستند. مقدار پیش فرض برای این ستون، pending در نظر گرفته شده که به معنای این است که آگهی در صف انتظار برای تعیین دسته بندی قرار دارد و به محض اتمام پردازش عکس و مراحل tagging، state جدید و category آگهی، مشخص شده و در جدول update می شود. در این فایل توابعی که برای مدیریت دیتابیس مورد نیاز هستند، پیاده سازی شده است. (مانند توابعی برای ایجاد ارتباط با دیتابیس، ایجاد یک سطر جدید در جدول به منظور ثبت اطلاعات آگهی جدید، دریافت اطلاعات مربوط به یک آگهی ثبت شده در جدول، دریافت وضعیت فعلی آگهی، به روزرسانی state و category آگهی)
S3_handler.py	هدف: مدیریت object storage (ذخیره ساز شیء) برای این بخش از سرویس ذخیره سازی ارائه شده توسط ابرآروان استفاده می شود. به این منظور یک bucket با نام ad.image ایجاد شده و تمامی عکس ها در این bucket ذخیره می شوند. لازم به ذکر است که عکس آگهی ها به نام id آن آگهی و در قالب {ad_id}.png ذخیره می شوند که بازیابی آن ها را به سادگی امکان پذیر می کند.

	این فایل دو function کلی را در بر دارد که شامل upload_to_s3_bucket و download_from_s3_bucket است. تابع اول وظیفه دارد فایل عکسی که توسط کاربر آپلود شده است را در S3 bucket قرار دهد و آن را در object storage ذخیره کند. تابع دوم عکس آگهی را با داشتن id آن، بازیابی و دانلود می کند. این تابع، هم در زمان پردازش عکس و هم در زمان نشان دادن state به کاربر در صورتی که آگهی در وضعیت accepted قرار داشته باشد، استفاده می شود.
image_processing.py	هدف: پردازش و برچسب زنی عکس آگهی و تعیین state و category جدید پردازش عکس توسط سرویس برچسب زنی Imagga انجام می شود. این سرویس یک عکس را دریافت می کند و پاسخ و نتیجه ی پردازش عکس را در قالب json بر می گرداند. این پاسخ شامل tag های عکس به همراه درصد اطمینان یا confidence هر کدام از آن tag ها است. در این مرحله باید در مورد قبول یا رد شدن آگهی تصمیم گرفته شود. اگر در میان tag ها برچسب "vehicle" موجود نبود یا درصد اطمینان آن از ۵۰ درصد کمتر بود این آگهی رد می شود (state = failed) و دسته بندی ای برای آن در نظر گرفته نخواهد شد. اما اگر این برچسب در میان آن ها موجود بود و درصد اطمینانی برابر یا بیشتر از ۵۰ درصد داشت، این آگهی پذیرفته می شود و اولین tag موجود در json (که بیشترین ضریب اطمینان را دارد) به عنوان دسته بندی آگهی انتخاب می شود. در مرحله ی پایانی هم state و هم category جدید در دیتابیس به روزرسانی می شوند.

- فایل های اصلی

- سرویس اول (Service-1.py)

این سرویس از دو API تشکیل شده است.

API اول برای ثبت یک آگهی جدید است و با زیردامنه ی "/ad_submission" قابل دسترسی است. با وارد کردن آدرس این دامنه و فرستادن درخواست GET، یک صفحه ی html (ad_submission.html) نمایش داده می شود که دارای مکان هایی برای وارد کردن اطلاعات آگهی کاربر (شامل ایمیل و description) و مکانی برای آپلود عکس موردنظر است.

با فشردن دکمه ی Submit یک request با متد POST ایجاد می شود. در این حالت ارتباط با دیتابیس برقرار شده و اطلاعات آگهی در پایگاه داده ذخیره می شود. همچنین id آگهی ثبت شده برگردانده می شود و با مشخص شدن id آگهی، تصویر آگهی در S3 bucket و با نام {ad_id}.png ذخیره می شود. علاوه بر آن، id آگهی به صف id_queue (صف ایجاد شده در سرویس CloudAMQP به منظور استفاده از RabbitMQ برای مدیریت آگهی های ثبت شده و انجام عملیات های پردازشی آن ها به ترتیب) اضافه می شود تا سرویس دوم عملیات پردازش عکس و ... را انجام دهد.

در نهایت صفحه ی html دیگری (ad_submission_result.html) نمایش داده خواهد شد که به کاربر از ثبت آگهی خود اطمینان می دهد و اطلاعات submit شده را نشان می دهد.

API دوم برای دریافت و بررسی state آگهی است و دسترسی به آن از طریق زیردامنه ی ''/check_state'' امکان پذیر است. با وارد کردن آدرس این دامنه و فرستادن درخواست GET، یک صفحه ی html (check_state.html) نمایش داده می شود که دارای مکانی برای وارد کردن id آگهی است.

بعد از وارد کردن id و فشردن دکمه ی Check، درخواستی از نوع POST ایجاد می شود که id وارد شده را دریافت می کند و بر اساس این id، state آن آگهی را از پایگاه داده تشخیص می دهد و در ادامه مطابق با دستور کار، پیام مناسب نمایش داده می شود.

اگر آگهی در وضعیت failed یا pending باشد، فایل های html متناسب با هر کدام از آن ها که دربردارنده ی پیام مناسب است نمایش داده می شود. (فایل های check_state_result_failed.html و check_state_result_pending.html)

اما اگر state آگهی، accepted باشد؛ ابتدا اطلاعات آگهی (شامل ایمیل و متن و دسته بندی) از پایگاه داده استخراج می شود و سپس عکس آگهی از S3 bucket دانلود می شود و در نهایت فایل html مربوط به این بخش که شامل همه ی اطلاعات آگهی به همراه عکس آن است نمایش داده می شود. (فایل check_state_result_accepted.html)

- سرویس دوم (Service-2.py)

این سرویس به صف RabbitMQ متصل شده و به پیام های جدید که حاوی id آگهی های جدید است گوش می دهد و با دریافت هر پیام، id آگهی را به صف id_queue اضافه می کند. سپس به صورت مداوم طول این صف را چک می کند و اگر id آگهی ای در آن وجود داشت، مراحل پردازش و بررسی آن آگهی را انجام می دهد (طبق ترتیب FIFO).

به این منظور، ابتدا عکس آگهی از S3 bucket دانلود می شود و در ادامه، برچسب زنی عکس و تعیین state و category با استفاده از توابع پیاده سازی شده در image_processing.py انجام می شود. سپس اتصال با دیتابیس برقرار شده و state و category آپدیت می شوند. همچنین آدرس ایمیل ثبت شده در این آگهی از دیتابیس دریافت می شود تا در گام بعد ارسال ایمیل انجام شود.

برای ارسال ایمیل از سرویس mailgun استفاده می شود که امکان ارسال ایمیل به آدرس های ایمیلی که به بخش Authorized Recipients اضافه و verify شده اند را می دهد. کد مربوط به ارسال ایمیل در تابع send_email پیاده سازی شده است که بر اساس state آگهی، متن ایمیل را مشخص کرده و آن را به ایمیل کاربر می فرستد.