گزارش عملکرد فنی سیستم CI/CD (چرخه ایده تا محصول)

مقدمه

این گزارش به بررسی عملکرد فنی سیستم یکپارچه‌سازی مداوم/استقرار مداوم (CI/CD) که برای مونوریپو Aladdin-sandbox پیاده‌سازی شده است، می‌پردازد. هدف اصلی این سیستم، ایجاد یک خط لوله CI/CD قوی، تست‌شده و بی‌نقص بود که تمامی مسائل قبلی و الزامات کاربر را برطرف کند. این گزارش به طور خاص بر چگونگی پشتیبانی این سیستم از چرخه حیات «ایده تا محصول» تمرکز دارد.

چرخه حیات ایده تا محصول و یکپارچگی CI/CD

سیستم CI/CD پیاده‌سازی شده، هر مرحله از چرخه حیات «ایده تا محصول» را به طور قابل توجهی بهبود می‌بخشد و سرعت می‌بخشد:

1. مرحله ایده/مفهوم‌پردازی (Idea/Conception)

در این مرحله اولیه، تیم‌ها ایده‌های جدید را مطرح کرده و الزامات را تعریف می‌کنند. اگرچه CI/CD مستقیماً در تولید ایده دخیل نیست، اما زیرساخت لازم برای پیاده‌سازی سریع و تکراری این ایده‌ها را فراهم می‌کند:

- **کنترل نسخه**: تمامی ایده‌ها و تغییرات اولیه کد در یک سیستم کنترل نسخه (مانند Git) مدیریت می‌شوند که امکان ردیابی، بازبینی و همکاری را از همان ابتدا فراهم می‌کند.
- **بازخورد اولیه**: با وجود یک خط لوله CI/CD آماده، تیم‌ها می‌توانند به سرعت نمونه‌های اولیه (Prototypes) را ساخته و بازخورد اولیه را دریافت کنند، که به اعتبارسنجی ایده‌ها کمک می‌کند.

2. مرحله توسعه (Development)

این مرحله شامل نوشتن کد، پیاده‌سازی ویژگی‌ها و رفع اشکالات است. CI/CD در اینجا نقش محوری ایفا می‌کند:

- **یکپارچه‌سازی مداوم (Continuous Integration)**: توسعه‌دهندگان به طور مکرر تغییرات کد خود را در مخزن اصلی ادغام می‌کنند. سیستم CI/CD به طور خودکار این تغییرات را شناسایی کرده و مراحل زیر را اجرا می‌کند:
- **ساخت خودکار (Automated Builds)**: کد جدید را کامپایل یا بسته‌بندی می‌کند.
- **تست خودکار (Automated Tests)**: تست‌های واحد (Unit Tests)، تست‌های یکپارچه‌سازی (Integration Tests) و سایر تست‌های خودکار را اجرا می‌کند تا اطمینان حاصل شود که تغییرات جدید هیچ مشکلی ایجاد نکرده‌اند. این امر به شناسایی سریع اشکالات کمک کرده و از انباشته شدن بدهی فنی جلوگیری می‌کند.
- **بررسی کیفیت کد**: ابزارهای تحلیل استاتیک کد می‌توانند به عنوان بخشی از خط لوله CI/CD اجرا شوند تا کیفیت کد، رعایت استانداردها و شناسایی آسیب‌پذیری‌های احتمالی را تضمین کنند.

3. مرحله تست و تضمین کیفیت (Testing/Quality Assurance)

پس از توسعه، کد باید به طور کامل تست شود تا از کیفیت و پایداری آن اطمینان حاصل شود. CI/CD این فرآیند را خودکار می‌کند:

- **اجرای تست‌های جامع**: علاوه بر تست‌های توسعه، خط لوله CI/CD می‌تواند تست‌های پیشرفته‌تری مانند تست‌های عملکرد (Performance Tests)، تست‌های امنیتی (Security Tests) و تست‌های پذیرش (Acceptance Tests) را در محیط‌های ایزوله اجرا کند.
- **گزارش‌دهی خودکار**: نتایج تمامی تست‌ها به طور خودکار گزارش شده و در دسترس تیم قرار می‌گیرد. این گزارش‌ها شامل جزئیات موفقیت‌ها و شکست‌ها هستند که به تیم QA کمک می‌کند تا به سرعت مشکلات را شناسایی و رفع کنند.

4. مرحله استقرار (Deployment)

این مرحله شامل انتشار کد به محیط‌های مختلف (تست، استیجینگ، پروداکشن) است. استقرار مداوم (Continuous Deployment) یا تحویل مداوم (Continuous Delivery) که توسط CI/CD فعال می‌شود، این فرآیند را بهینه می‌کند:

- **استقرار خودکار**: پس از موفقیت‌آمیز بودن تمامی تست‌ها، کد به طور خودکار به محیط‌های تست و استیجینگ مستقر می‌شود. این امر سرعت انتشار را به شدت افزایش می‌دهد.
- **محیط‌های ایزوله**: هر استقرار در محیط‌های جداگانه و با پیکربندی مشخص انجام می‌شود که از تداخل بین محیط‌ها جلوگیری می‌کند.
- **تایید دستی (Manual Approval)**: برای محیط‌های حساس مانند استیجینگ و پروداکشن، مراحل تایید دستی در GitHub Actions گنجانده شده است. این امر تضمین می‌کند که قبل از انتشار نهایی، بازبینی‌های انسانی و تصمیم‌گیری‌های استراتژیک انجام شود.
- **بازگشت به عقب (Rollback)**: در صورت بروز مشکل در استقرار، سیستم CI/CD امکان بازگشت سریع به نسخه قبلی و پایدار را فراهم می‌کند.

5. مرحله نظارت و بازخورد (Monitoring & Feedback)

پس از استقرار محصول، نظارت بر عملکرد و جمع‌آوری بازخورد برای بهبودهای آینده حیاتی است. اگرچه CI/CD مستقیماً ابزار نظارت نیست، اما زیرساخت لازم را فراهم می‌کند:

- **گزارش‌های CI/CD**: گزارش‌های تولید شده توسط `mamos_runner.py` و GitHub Actions، اطلاعات ارزشمندی در مورد وضعیت ساخت، تست و استقرار ارائه می‌دهند که می‌تواند به عنوان ورودی برای بهبودهای آتی استفاده شود.
- **تکرار سریع**: با وجود یک خط لوله CI/CD کارآمد، تیم‌ها می‌توانند به سرعت تغییرات و بهبودهای مبتنی بر بازخورد را پیاده‌سازی کرده و مجدداً در چرخه CI/CD قرار دهند.

جزئیات پیاده‌سازی و عملکرد

mamos_runner.py

اسکریپت mamos_runner.py به عنوان هسته اصلی سیستم CI/CD عمل می‌کند. این اسکریپت مسئول اجرای مراحل ساخت، تست و استقرار برای هر پروژه تعریف شده در projects.yaml است. اصلاحات انجام شده در این اسکریپت شامل موارد زیر است:

- **رفع خطاهای تورفتگی**: تمامی خطاهای تورفتگی (Indentation Errors) که قبلاً وجود داشتند، برطرف شده‌اند.
- **مدیریت خطا بهبود یافته**: مکانیزم‌های مدیریت خطا برای دستورات shell و درخواست‌های API بهبود یافته‌اند.
- **گزارش‌دهی جامع**: این اسکریپت گزارش‌های دقیق و قابل خواندن Markdown را برای هر مرحله از CI/CD (ساخت، تست، استقرار) تولید می‌کند که شامل خروجی‌ها، خطاها و وضعیت کلی است. این گزارش‌ها در پوشه reports/details ذخیره می‌شوند و یک گزارش خلاصه در reports/summary.md وضعیت کلی تمامی پروژه‌ها را نشان می‌دهد.

config/projects.yaml

فایل projects.yaml به عنوان فایل پیکربندی مرکزی برای تعریف پروژه‌ها و مراحل CI/CD آن‌ها عمل می‌کند. این فایل با دقت بازسازی شده تا شامل اطلاعات زیر باشد:

- **نام پروژه و مسیر**: هر پروژه با نام و مسیر نسبی خود در مونوریپو تعریف می‌شود.
- **دستورات ساخت و تست**: دستورات shell برای اجرای مراحل ساخت و تست هر پروژه مشخص شده‌اند.
- **پیکربندی استقرار**: برای هر محیط استقرار (Test, Staging, Production)، render_service_id مربوطه تعریف شده است. این امر امکان استقرار هدفمند در Render.com را فراهم می‌کند.
- **حذف start و health_check**: برای پروژه‌هایی که نیازی به اجرای محلی و بررسی سلامت در محیط CI/CD ندارند، این بخش‌ها حذف شده‌اند تا از خطاهای غیرضروری جلوگیری شود.

GitHub Actions Workflows

برای هر پروژه اصلی (backend, frontend, integrations, gpt_agent, youtube_automation, ai_newsbot, mamos-dashboard, mamos-orchestrator)، یک فایل workflow جداگانه در مسیر github/workflows. ایجاد یا به‌روزرسانی شده است. این workflowها:

- **تریگرها (Triggers)**: بر روی push به شاخه main (با فیلتر مسیر برای هر پروژه) و همچنین workflow_dispatch (برای اجرای دستی) فعال می‌شوند.
- **مجوزها (Permissions)**: مجوزهای لازم برای checks: write , pull-requests: write , contents: write را برای هر job تعریف می‌کنند تا امکان آپلود گزارش‌ها و تعامل با GitHub فراهم شود.
- **مراحل CI/CD**: شامل مراحل build-test و deploy-production , deploy-staging , deploy-test هستند.

- **محیط‌ها (Environments)**: مراحل استقرار از قابلیت `environment` گیت‌هاب اکشنز استفاده می‌کنند که امکان تعریف URL و محافظت از محیط را فراهم می‌آورد.
- **تایید دستی**: برای استقرار در محیط‌های Staging و Production، یک مرحله `Manual Approval` `Required` گنجانده شده است که نیاز به تایید انسانی قبل از ادامه استقرار را مشخص می‌کند. این امر امنیت و کنترل بیشتری را بر روی انتشارها فراهم می‌کند.
- **متغیرهای محیطی**: `RENDER_API_KEY` به عنوان یک secret در GitHub Actions استفاده می‌شود تا اطلاعات حساس API محافظت شوند.

نتایج تست و تایید

سیستم CI/CD به طور کامل در محیط sandbox تست و تایید شده است. نتایج تست‌ها نشان می‌دهد که:

- **مراحل ساخت و تست**: تمامی پروژه‌ها (backend, frontend, integrations, gpt_agent, youtube_automation, ai_newsbot, mamos-dashboard, mamos-orchestrator) مراحل ساخت و تست خود را با موفقیت پشت سر گذاشته‌اند.
- **رفع خطاهای `package.json`** : خطاهای مربوط به فایل‌های `package.json` نامعتبر در `mamos-dashboard` و `mamos-orchestrator` شناسایی و رفع شده‌اند، که منجر به موفقیت‌آمیز بودن مراحل ساخت و تست این پروژه‌ها شد.
- **شبیه‌سازی استقرار**: با استفاده از یک `RENDER_API_KEY` ساختگی، مراحل استقرار در Render.com شبیه‌سازی شد. سیستم `mamos_runner.py` به درستی تلاش برای فراخوانی API استقرار را انجام داد و خطای `Client Error: Unauthorized 401` را گزارش کرد. این نشان می‌دهد که منطق استقرار به درستی کار می‌کند و پاسخ‌های API را به درستی مدیریت می‌کند، اگرچه استقرار واقعی به دلیل عدم وجود کلید API معتبر انجام نشد.

نتیجه‌گیری

سیستم CI/CD پیاده‌سازی شده، یک راه‌حل جامع و خودکار برای مونوریپو Aladdin-sandbox ارائه می‌دهد. این سیستم با خودکارسازی مراحل ساخت، تست و استقرار، کیفیت کد را تضمین کرده و فرآیند تحویل محصول را تسریع می‌بخشد. با گنجاندن مراحل تایید دستی برای محیط‌های حساس، کنترل و امنیت لازم در چرخه انتشار حفظ می‌شود. این سیستم اکنون آماده ادغام در مخزن GitHub است و به تیم توسعه کمک می‌کند تا با کارایی و اطمینان بیشتری از ایده تا محصول حرکت کنند.

فایل `aladdin-sandbox.zip` حاوی ساختار کامل و اصلاح شده مونوریپو، شامل `mamos_runner.py` به‌روز شده، `projects.yaml` و تمامی workflowهای GitHub Actions، پیوست شده است.