مدیریت تست، تخصیص منابع و تنظیمات برای چندین محصول در سیستم CI/CD

مقدمه

در یک مونوریپو که شامل چندین محصول (پروژه) است، مدیریت کارآمد ترتیب تست، تخصیص منابع و تنظیمات برای هر پروژه از اهمیت بالایی برخوردار است. سیستم CI/CD پیاده‌سازی شده برای aladdin-sandbox با استفاده از mamos_runner.py و GitHub Actions Workflows، این جنبه‌ها را به صورت متمرکز و خودکار مدیریت می‌کند.

1. ترتیب تست و اجرای پروژه‌ها

سیستم CI/CD ما به گونه‌ای طراحی شده است که انعطاف‌پذیری لازم برای اجرای تست‌ها را فراهم کند:

- **اجرای ترتیبی (Sequential Execution)**: به صورت پیش‌فرض، mamos_runner.py پروژه‌ها را به ترتیبی که در فایل config/projects.yaml تعریف شده‌اند، پردازش می‌کند. این بدان معناست که برای هر پروژه، ابتدا مرحله build و سپس مرحله test اجرا می‌شود و پس از اتمام کامل این مراحل برای یک پروژه، به سراغ پروژه بعدی می‌رود. این رویکرد برای اطمینان از عدم تداخل و مدیریت ساده‌تر وابستگی‌ها (در صورت وجود) مفید است.
- **اجرای موازی (Parallel Execution) از طریق GitHub Actions**: اگرچه mamos_runner.py به صورت ترتیبی پروژه‌ها را پردازش می‌کند، اما GitHub Actions این قابلیت را دارد که چندین workflow را به صورت موازی اجرا کند. از آنجایی که برای هر پروژه اصلی (مانند backend , frontend , youtube_automation) یک فایل workflow جداگانه (github/workflows/*.yml.) ایجاد شده است، هر یک از این workflowها می‌توانند به صورت مستقل و همزمان توسط GitHub Actions اجرا شوند. این امر به طور موثری تست و بیلد چندین پروژه را به صورت موازی انجام می‌دهد و زمان کلی اجرای CI/CD را کاهش می‌دهد.
- **تریگرهای مبتنی بر تغییرات (Path-based Triggers)**: هر workflow در GitHub Actions با استفاده از paths در تریگر on: push: پیکربندی شده است. این بدان معناست که تنها در صورتی که تغییراتی در مسیر مربوط به یک پروژه خاص (مثلاً **/apps/backend برای پروژه backend) ایجاد شود، workflow مربوط به آن پروژه فعال می‌شود. این بهینه‌سازی از اجرای غیرضروری CI/CD برای پروژه‌هایی که تغییری نکرده‌اند، جلوگیری می‌کند و منابع را حفظ می‌کند.
- **اجرای انتخابی (Selective Execution)**: با استفاده از workflow_dispatch در GitHub Actions، می‌توان به صورت دستی یک workflow خاص را برای یک پروژه مشخص اجرا کرد. این قابلیت به توسعه‌دهندگان اجازه می‌دهد تا تنها پروژه‌ای را که روی آن کار می‌کنند، تست یا مستقر کنند، بدون اینکه نیاز به اجرای کامل CI/CD برای تمامی پروژه‌ها باشد.

2. تخصیص منابع

تخصیص منابع در سیستم CI/CD ما عمدتاً توسط GitHub Actions و محیط اجرایی آن مدیریت می‌شود:

- **رانرهای GitHub Actions**: هر workflow در GitHub Actions بر روی یک رانر (Runner) اجرا می‌شود. GitHub Actions رانرهای میزبانی شده (hosted runners) را فراهم می‌کند که منابع محاسباتی (CPU، RAM) را به صورت پویا برای اجرای Jobها تخصیص می‌دهند. اگر چندین workflow به صورت موازی اجرا شوند، GitHub Actions به طور خودکار رانرهای جدیدی را برای مدیریت بار کاری فراهم می‌کند (تا سقف محدودیت‌های حساب کاربری).
- **ایزوله‌سازی Jobها**: هر Job در یک workflow در یک محیط ایزوله و تمیز اجرا می‌شود. این ایزوله‌سازی تضمین می‌کند که وابستگی‌ها و تنظیمات یک پروژه بر روی پروژه دیگر تأثیر نمی‌گذارد و نتایج تست‌ها قابل اعتماد هستند.
- **مدیریت وابستگی‌های پایتون/Node.js**: در مرحله `build` هر پروژه، دستوراتی مانند `pip install -r requirements.txt` یا `npm install` اجرا می‌شوند. این دستورات وابستگی‌های خاص هر پروژه را در محیط ایزوله رانر نصب می‌کنند، بنابراین هر پروژه منابع مورد نیاز خود را به صورت مستقل مدیریت می‌کند.

3. تنظیمات و پیکربندی

تنظیمات برای چندین محصول به صورت متمرکز و ساختاریافته مدیریت می‌شوند تا از سازگاری و سهولت نگهداری اطمینان حاصل شود:

- **پیکربندی مرکزی پروژه (`config/projects.yaml`)**:
  - این فایل YAML به عنوان منبع اصلی حقیقت (Single Source of Truth) برای تعریف تمامی پروژه‌ها در مونوریپو عمل می‌کند. هر ورودی در این فایل شامل `name`، `path`، `build` command، `test` command و پیکربندی `deploy` برای محیط‌های مختلف (Test, Staging, Production) است.
  - این رویکرد تضمین می‌کند که تمامی اطلاعات مربوط به نحوه ساخت، تست و استقرار یک پروژه در یک مکان متمرکز و قابل خواندن توسط انسان قرار دارد.
- **پیکربندی فرآیند (GitHub Actions Workflows)**:
  - فایل‌های `github/workflows/*.yml.` نحوه اجرای CI/CD را برای هر پروژه تعریف می‌کنند. این فایل‌ها شامل تریگرها، مجوزها، مراحل (steps) و محیط‌های استقرار هستند.
  - این workflowها از `mamos_runner.py` استفاده می‌کنند و پارامترهای لازم (مانند `--project` و `--deploy-env`) را به آن ارسال می‌کنند. این جداسازی مسئولیت‌ها (پیکربندی پروژه در `projects.yaml` و پیکربندی فرآیند در workflowها) باعث افزایش خوانایی و نگهداری سیستم می‌شود.
- **متغیرهای محیطی و Secrets**: اطلاعات حساس مانند `RENDER_API_KEY` به عنوان Secrets در GitHub Actions ذخیره می‌شوند و به صورت متغیرهای محیطی در زمان اجرای workflow در دسترس قرار می‌گیرند. این امر امنیت اطلاعات حساس را تضمین می‌کند و از قرار گرفتن آن‌ها در کد یا فایل‌های پیکربندی عمومی جلوگیری می‌کند.

نتیجه‌گیری

سیستم CI/CD پیاده‌سازی شده، با استفاده از ترکیب هوشمندانه `mamos_runner.py` برای ارکستراسیون داخلی پروژه و GitHub Actions برای اتوماسیون و مدیریت منابع در سطح مونوریپو، قادر است ترتیب تست، تخصیص منابع و تنظیمات را برای چندین محصول به صورت کارآمد و مقیاس‌پذیر مدیریت کند. این رویکرد امکان توسعه موازی، تست‌های ایزوله و استقرارهای کنترل شده را فراهم می‌آورد که برای یک مونوریپو با محصولات متعدد ضروری است.