

تحقیق برنامه نویسی وب

نقش داکر در توسعه و استقرار برنامه های وب
یک مثال عملی و بارگزاری ویدیو آن در آپارات +

سید محمد پویان شمس الدین

401110812

آریان اکبری

محمد شفیع زاده

401110386

داکر به عنوان یک پلتفرم متن‌باز برای کانتینرسازی، توانسته است فرآیند توسعه، تست و استقرار نرم‌افزارها را بسیار بهبود ببخشد. با استفاده از این ابزار، برنامه‌ها به صورت بسته‌های ایزوله (کانتینر) همراه با تمامی وابستگی‌ها و تنظیمات لازم اجرا می‌شوند؛ به‌طوری که مشکلات ناشی از تفاوت‌های محیطی بین توسعه، تست و تولید به حداقل می‌رسد. این امر باعث افزایش سرعت انتشار و بهبود کیفیت نهایی برنامه‌های وب شده است.

داکر به عنوان یک ابزار قدرتمند کانتینری‌سازی، نقش زیادی در بهبود فرآیندهای توسعه و استقرار برنامه‌های وب دارد. با فراهم کردن محیط‌های ایزوله، قابل حمل و یکپارچه، داکر تضمین می‌کند که برنامه‌ها در هر مرحله از چرخه توسعه از توسعه اولیه تا استقرار در محیط‌های تولید به همان شکل اجرا شوند. این ویژگی‌ها موجب کاهش خطاها، افزایش سرعت استقرار و بهبود بهره‌وری منابع می‌شود.

داکر برخی چالش‌ها از جمله مسائل امنیتی و مدیریت داده‌های پایدار دارد، با استفاده از ابزارهای مکمل مانند Docker Compose و Kubernetes می‌توان این چالش‌ها را برطرف کرد. در نهایت، به کارگیری داکر در پروژه‌های وب به تیم‌های توسعه و عملیات کمک می‌کند تا نرم‌افزارهایی مقیاس‌پذیر، پایدار و کارآمد ارائه دهند.

۲. مفاهیم پایه‌ای داکر

الف) کانتینرها

کانتینرها محیط‌های اجرایی ایزوله‌ای هستند که در آن‌ها برنامه به همراه کتابخانه‌ها، فایل‌های باینری و تنظیمات موردنیازش بسته‌بندی می‌شود. این ساختار به توسعه‌دهندگان اجازه می‌دهد تا بدون نگرانی از ناسازگاری‌های محیطی، برنامه‌های خود را در هر زیرساختی (لینوکس، ویندوز یا ابری) اجرا کنند.

ب) Docker Image و Dockerfile

- **داکر ایمج:** یک قالب ثابت است که شامل تمام اطلاعات لازم برای ایجاد یک کانتینر می‌باشد. ایمج‌ها به‌عنوان الگو عمل کرده و امکان ساخت کانتینرهای متعدد از یک منبع واحد را فراهم می‌کنند.
- **داکر فایل:** یک اسکریپت متنی است که دستورالعمل‌های لازم برای ساخت یک ایمج را تعریف می‌کند. این فایل به صورت خودکار و تکرارپذیر ساخت تصویر را تضمین می‌کند و باعث می‌شود فرآیند تولید محیط‌های اجرایی استاندارد شود.

ج) ابزارهای مکمل

ابزارهایی مانند **Docker Compose** این امکان را می‌دهد که چندین کانتینر مرتبط (مثلاً سرور وب، پایگاه داده و کش) را به صورت همزمان و با یک فایل پیکربندی (YAML) مدیریت و اجرا کنند. این ابزارها در ایجاد محیط‌های چندبخشی و میکروسرویسی کاربرد دارد

۳. نقش داکر در توسعه و استقرار برنامه‌های وب

الف) ایجاد محیط‌های توسعه یکپارچه

یکی از بزرگ‌ترین چالش‌ها در توسعه برنامه‌های وب، اختلاف محیط‌های توسعه، تست و تولید است. داکر با بسته‌بندی تمامی وابستگی‌های لازم در یک کانتینر، تضمین می‌کند که همان محیط یکپارچه و قابل انتقال در تمامی مراحل چرخه عمر نرم‌افزار فراهم شود. این ویژگی باعث کاهش خطاها و افزایش سرعت توسعه می‌شود.

ب) تسهیل استقرار و مهاجرت

با توجه به قابلیت حمل (Portability) کانتینرها، برنامه‌های وب می‌توانند به راحتی از یک سیستم به سیستم دیگر (مانند انتقال از محیط تست به سرورهای ابری) منتقل شوند. این امر به ویژه در شرکت‌هایی که نیاز به استقرار سریع و مقیاس‌پذیر دارند، بسیار مهم است.

ج) مدیریت میکروسرویس‌ها

داکر با فراهم ساختن کانتینرهای مجزا، مدیریت و هماهنگی بین این سرویس‌ها را ساده می‌کند و توسعه‌دهندگان می‌توانند بدون نگرانی از تداخل سرویس‌ها، بر بهبود عملکرد و کارایی تمرکز کنند.

د) ادغام با فرآیندهای CI/CD

داکر به عنوان بخشی از چرخه توسعه مدرن (DevOps) نقش کلیدی در خودکارسازی فرآیندهای توسعه و استقرار دارد. با ادغام ابزارهایی مانند Jenkins، GitLab CI و سایر سیستم‌های CI/CD، تغییرات در کد به سرعت در کانتینرهای آزمایشی اجرا و پس از تایید، به محیط تولید منتقل می‌شود. این فرآیند باعث بهبود کیفیت نرم‌افزار و کاهش زمان انتشار نسخه ی جدید می‌شود.

۴. مزایا و چالش‌های استفاده از داکر

مزایای داکر:

1. **قابلیت حمل بالا:** برنامه‌ها در کانتینرهایی بسته‌بندی می‌شوند که در هر محیطی (سیستم‌های مختلف، سرورهای ابری) قابل اجرا هستند.
2. **اجرای ایزوله:** هر کانتینر به صورت مجزا اجرا می‌شود؛ بنابراین، خطا یا مشکل در یکی از آن‌ها بر سایر برنامه‌ها تأثیری ندارد.
3. **سرعت بالا و بهره‌وری منابع:** نسبت به ماشین‌های مجازی، کانتینرها سریع‌تر راه‌اندازی شده و از منابع سیستم به صورت بهینه‌تری استفاده می‌کنند.
4. **مقیاس‌پذیری آسان:** به راحتی می‌توان تعداد کانتینرها را بر اساس نیاز افزایش یا کاهش داد.
5. **سهولت در پیاده‌سازی CI/CD:** ادغام با سیستم‌های خودکارسازی توسعه، روند استقرار برنامه‌ها را تسریع می‌کند.
6. **مدیریت وابستگی‌ها:** تمامی وابستگی‌های نرم‌افزار در ایمج ذخیره می‌شود که احتمال بروز مشکلات ناشی از اختلاف نسخه‌ها را کاهش می‌دهد.
7. **کاهش هزینه‌های زیرساختی:** بهینه‌سازی مصرف منابع موجب کاهش هزینه‌های اجرایی سرورها می‌شود.
8. **همکاری تیمی:** امکان به اشتراک‌گذاری ایمج‌ها از طریق سرویس‌هایی مانند Docker Hub، همکاری بین تیم‌ها را تسهیل می‌کند.
9. **پایداری و ایزوله‌سازی:** مشکلات یک کانتینر تأثیری بر بقیه نخواهد داشت.
10. **افزایش سرعت استقرار:** امکان استقرار سریع و بدون وقفه برنامه‌ها.

چالش‌های داکر:

- **پیچیدگی مدیریت کانتینرها:** در پروژه‌های بزرگ ممکن است مدیریت تعداد زیادی کانتینر چالش‌برانگیز باشد.
- **مسائل امنیتی:** اگر به درستی پیکربندی نشود، احتمال دسترسی غیرمجاز به منابع وجود دارد؛ بنابراین نیاز به رعایت نکات امنیتی دارد.
- **چالش‌های ذخیره‌سازی پایدار:** مدیریت داده‌های پایدار در کانتینرها ممکن است پیچیده باشد و نیاز به راهکارهای اضافی مثل Volume ها دارد.
- **وابستگی به ابزارهای مکمل:** برای مدیریت در مقیاس بزرگ، نیاز به ابزارهایی مانند Kubernetes حس می‌شود که این امر به پیچیدگی‌های فنی بیشتری منجر می‌شود.

بخش امتیازی:

در این قسمت به ویدیو در رابطه با این تحقیق و مثال عملی مرتبط در رسانه آپارات بارگزاری کرده ایم.

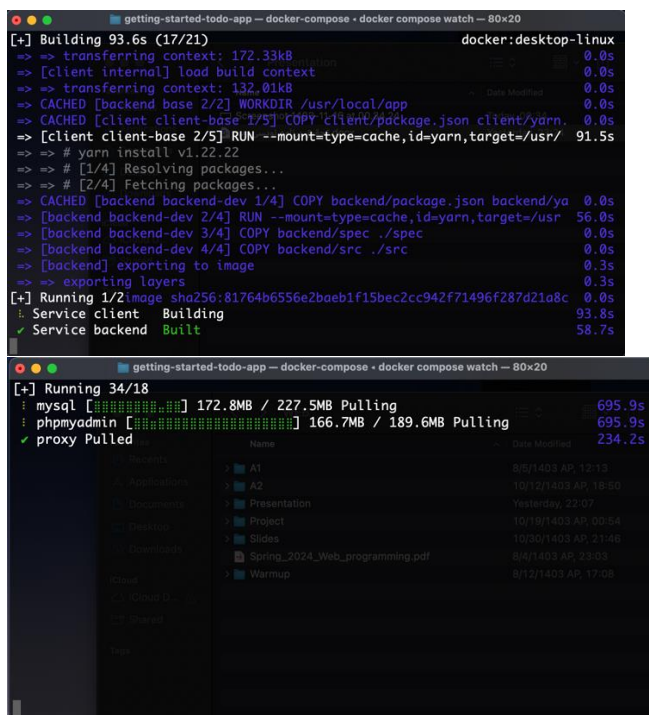
لینک ویدیو ضبط شده : <https://www.aparat.com/v/uyt2ntx>

نحوه اجرا

git clone <https://github.com/docker/getting-started-todo-app>

cd getting-started-todo-app

docker compose watch



The first screenshot shows the output of the 'docker compose watch' command. It displays the build process for the 'getting-started-todo-app' service. The build is successful, and the image is exported to the Docker registry. The second screenshot shows the output of the 'docker compose watch' command after the build. It displays the status of the services: 'mysql' is pulling, 'phpmyadmin' is pulling, and 'proxy' is pulled. The 'proxy' service is also pulling.

نحوه پوش در داکر هاب

docker build -t <DOCKER_USERNAME>/getting-started-todo-app .

docker image ls

```
fraxea@Mohammads-MacBook-Air-4 getting-started-todo-app % docker image ls
REPOSITORY                                TAG          IMAGE ID      CREATED        SIZE
fraxea/getting-started-todo-app            latest       9c8d80d8fc90  49 seconds ago 1.12GB
getting-started-todo-app-client            latest       964e6d626a37  28 minutes ago 1.19GB
getting-started-todo-app-backend           latest       81764b6556e2  38 minutes ago 1.17GB
flask-app                                  latest       4bf543dd49b7  2 hours ago    1.03GB
traefik                                     v2.11        b4e324f1e3ca  3 days ago     174MB
phpmyadmin                                  latest       c2ceb62da5b0  10 days ago    581MB
mysql                                       8.0          28a9ebb54767  13 days ago    760MB
```

`docker push <DOCKER_USERNAME>/getting-started-todo-app`

The screenshot shows the Docker Hub interface for the repository `fraxea/getting-started-todo-app`. The page is divided into several sections:

- Repository Header:** Shows the repository name, a lock icon, and the last push time: "Last pushed less than a minute ago · Repository size: 379.4 MB".
- Short Description:** A section for a short description and adding a category.
- Tags:** A table showing the repository contains 1 tag(s). The tag is `latest`, pushed 5 minutes ago.
- Docker commands:** A section showing the command to push a new tag: `docker push fraxea/getting-started-todo-app:tagname`.
- Automated builds:** A section explaining how to connect to GitHub or Bitbucket for automated builds.
- Repository overview:** A section for an overview of the repository, with an "Add overview" button.

منابع:

<https://docs.docker.com/>

<https://expressjs.com/>

<https://www.geeksforgeeks.org/introduction-to-docker/>

<https://www.ibm.com/think/topics/docker>

<https://docs.docker.com/get-started/docker-overview/>

<https://www.papertrail.com/solution/guides/docker-everything-you-need-to-know/>