

Mediu Docker pentru aplicație și bază

Fișă publică de proiect · exerciții și criterii de acceptare

Construiești un mediu cu două servicii, volum persistent și rețea internă. Proiectul explică limitele containerelor și separarea de configurația de producție.

Livrabile

- Dockerfile
- compose.yaml
- Volum
- Health check
- README

Criterii de acceptare

- Baza nu este expusă inutil
- Containerul rulează fără root unde este posibil
- Secretele nu intră în imagine

Containerul ca mediu de rulare, nu mașină completă

O imagine descrie sistemul de fișiere și configurația de pornire, iar containerul este o instanță de execuție. Containerele împart anumite componente ale gazdei și nu trebuie tratate automat ca mașini virtuale independente. Începe cu o comandă care nu expune porturi și nu montează date private. Verifică identitatea imaginii și versiunea folosită. Pentru reproductibilitate, un tag poate necesita fixare mai strictă după politica proiectului.

Exemplu de lucru

```
docker run --rm hello-world
docker image ls
docker ps -a
```

Întrebări: ce imagine a fost descărcată; ce proces a rulat; ce s-a păstrat după terminare?
Nu adăuga: --privileged sau montarea rădăcinii gazdei pentru un exemplu simplu.

Interpretarea rezultatului

--rm elimină containerul terminat, nu toate imaginile sau volumele. Docker necesită un mediu instalat și autorizat pe mașina de laborator. Accesul la daemon poate conferi drepturi importante asupra gazdei; nu îl distribui necontrolat.

Exercițiu de consolidare

Explică ce diferență observi între imaginea listată și containerul care nu mai rulează.

1. Ce rezolvă Docker

Explică ce diferență observi între imaginea listată și containerul care nu mai rulează.

Verificare orientativă

--rm elimină containerul terminat, nu toate imaginile sau volumele. Docker necesită un mediu instalat și autorizat pe mașina de laborator. Accesul la daemon poate conferi drepturi importante asupra gazdei; nu îl distribui necontrolat.

De evitat

Un container nu este o mașină virtuală completă.

Imagine fixată și proces de container observabil

Pornește un container cu nume și port local explicit, apoi verifică logurile și starea. Nu expune un serviciu pe toate interfețele dacă este doar pentru laborator. O imagine descărcată poate fi actualizată sub același tag; pentru reproducere strictă, verifică digestul. Nu folosi latest ca dovadă că rulezi exact aceeași versiune ca alt coleg. Datele scrise în stratul containerului nu sunt un mecanism de persistență pe termen lung.

Exemplu de lucru

```
docker run -d --name web-demo -p 127.0.0.1:8088:80 nginx:stable-alpine
docker ps
docker logs web-demo
Adresă: http://127.0.0.1:8088/
Oprire controlată a containerului de laborator: docker stop web-demo
Nu aplica operații globale de prune pentru acest exercițiu.
```

Interpretarea rezultatului

Portul gazdei 8088 este mapat la portul 80 din container. Bind-ul 127.0.0.1 limitează accesul local. Statusul running nu garantează conținutul corect; verifică și răspunsul HTTP. Pentru un proiect reproductibil, înregistrează digestul imaginii efectiv folosite.

Exercițiu de consolidare

Pornește al doilea container cu alt nume și port, apoi identifică separat logurile fiecăruia.

2. Imagini și containere

Pornește al doilea container cu alt nume și port, apoi identifică separat logurile fiecăruia.

Verificare orientativă

Portul gazdei 8088 este mapat la portul 80 din container. Bind-ul 127.0.0.1 limitează accesul local. Statusul running nu garantează conținutul corect; verifică și răspunsul HTTP. Pentru un proiect reproductibil, înregistrează digestul imaginii efectiv folosite.

De evitat

Nu păstra singura copie a bazei în stratul efemer.

Persistență și rețea ca două probleme distincte

Un volum păstrează date în afara ciclului de viață al containerului, iar rețeaua permite comunicarea între servicii. Nu expune baza de date pe gazdă dacă aplicația o poate accesa pe rețeaua internă. Un volum nu este backup: o ștergere logică sau corupție poate rămâne în el. Verifică persistența prin recrearea containerului în laborator și backupul prin restaurare într-un volum separat.

Exemplu de lucru

Arhitectură demo: web → rețea internă → db.
Volum: db_data pentru datele DB.
Port public: numai web, dacă este necesar.
Teste: recreare container DB păstrează datele; volum nou începe separat; aplicația rezolvă numele serviciului DB.
Nu confunda: localhost din container cu gazda sau alt container.

Interpretarea rezultatului

Fiecare container are propriul context de rețea. În Compose, numele serviciului poate fi folosit pentru comunicarea internă conform configurației. Nu monta același director de date DB în două instanțe active fără suportul explicit al motorului.

Exercițiu de consolidare

Describe diferența dintre recrearea containerului și ștergerea volumului și ce dovadă ar confirma recuperarea.

3. Volume și rețele

Describe diferența dintre recrearea containerului și ștergerea volumului și ce dovadă ar confirma recuperarea.

Verificare orientativă

Fiecare container are propriul context de rețea. În Compose, numele serviciului poate fi folosit pentru comunicarea internă conform configurației. Nu monta același director de date DB în două instanțe active fără suportul explicit al motorului.

De evitat

Publicarea portului bazei nu este necesară pentru comunicarea internă.

Dockerfile cu dependențe și fișiere controlate

Un Dockerfile trebuie să declare baza, directorul de lucru, dependențele și comanda. Nu copia .env, chei sau directoare inutile în imagine. .dockerignore limitează contextul trimis la build. Pentru un exemplu static, folosește o imagine web și un fișier public; pentru PHP, extensiile și procesul de servire trebuie proiectate separat. Nu presupune că un exemplu de nginx execută PHP.

Exemplu de lucru

```
Dockerfile demo static:
FROM nginx:stable-alpine
COPY public/ /usr/share/nginx/html/
```

```
.dockerignore:
.git
.env
node_modules
vendor
```

```
Build: docker build -t atelier-static:demo .
Rulare: port local explicit, ca în lecția anterioară.
```

Interpretarea rezultatului

Imaginea conține numai fișierele publice copiate, dacă structura este respectată. Verifică lipsa secretelor înainte de publicarea imaginii într-un registry. Pentru versiuni exacte, fixează baza după politica proiectului și reconstruiește când sunt necesare actualizări de securitate.

Exercițiu de consolidare

Adaugă un fișier privat în afara public și confirmă că nu este inclus de COPY-ul ales.

4. Dockerfile pentru aplicația ta

Adaugă un fișier privat în afara public și confirmă că nu este inclus de COPY-ul ales.

Verificare orientativă

Imaginea conține numai fișierele publice copiate, dacă structura este respectată. Verifică lipsa secretelor înainte de publicarea imaginii într-un registry. Pentru versiuni exacte, fixează baza după politica proiectului și reconstruiește când sunt necesare actualizări de securitate.

De evitat

Fișierele șterse într-un layer ulterior pot rămâne în imagine.

Compose cu servicii și volum declarate

Compose descrie o aplicație cu mai multe servicii, dar nu face automat configurația sigură pentru producție. Pentru laborator, un serviciu web poate servi fișiere readonly pe un port local. Într-un stack cu DB, definește volume, rețele și health checks potrivite. `depends_on` nu trebuie interpretat ca dovadă universală că serviciul este gata să accepte operații. Aplicația trebuie să trateze și indisponibilitatea temporară a dependențelor.

Exemplu de lucru

```
compose.yaml demo static:
services:
  web:
    image: nginx:stable-alpine
    ports:
      - "127.0.0.1:8089:80"
    volumes:
      - ./public:/usr/share/nginx/html:ro
```

Verificare: `docker compose config`

Pornire: `docker compose up -d`

Observare: `docker compose ps`; `docker compose logs web`

Interpretarea rezultatului

Montarea readonly împiedică scrierea prin acel mount, dar nu este o izolare completă a containerului. Verifică directorul public și nu monta rădăcina proiectului dacă include secrete. Comanda `config` validează forma și substituțiile, nu funcționarea aplicației.

Exercițiu de consolidare

Adaugă un health check potrivit serviciului și explică diferența dintre container pornit și serviciu sănătos.

5. docker compose

Adaugă un health check potrivit serviciului și explică diferența dintre container pornit și serviciu sănătos.

Verificare orientativă

Montarea readonly împiedică scrierea prin acel mount, dar nu este o izolare completă a containerului. Verifică directorul public și nu monta rădăcina proiectului dacă include secrete. Comanda config validează forma și substituțiile, nu funcționarea aplicației.

De evitat

Ordinea pornirii nu garantează disponibilitatea serviciului.

Imagine de producție fără instrumente inutile

Un container de producție trebuie să aibă proces clar, semnale de oprire gestionate și permisiuni minime. Nu include dependențe de dezvoltare sau secrete în straturile imaginii. Un build multi-stage poate separa compilarea de runtime, dar trebuie verificat că rezultatul include toate fișierele necesare. Logurile se colectează prin mecanismul infrastructurii, iar datele persistente rămân în volume sau servicii dedicate. Nu stoca uploaduri numai în stratul containerului.

Exemplu de lucru

Checklist: bază verificată; dependențe runtime; utilizator potrivit; fișiere readonly unde se poate; health check; oprire controlată; limite resurse; secrete la rulare; backup pentru date. Test: recreare container, încărcare asseturi, acces storage, joburi și oprire în timpul unei operații de laborator.

Interpretarea rezultatului

Un utilizator non-root reduce anumite riscuri, dar nu repară o aplicație vulnerabilă sau un mount prea larg. Verifică ce trebuie să fie scriibil și de ce. Nu transforma toată imaginea în 777 pentru a rezolva o permisiune de storage.

Exercițiu de consolidare

Identifică directoarele scriibile pentru o aplicație Laravel și separă-le de codul care ar trebui să rămână nemodificat.

6. Container de producție

Identifică directoarele scriibile pentru o aplicație Laravel și separă-le de codul care ar trebui să rămână nemodificat.

Verificare orientativă

Un utilizator non-root reduce anumite riscuri, dar nu repară o aplicație vulnerabilă sau un mount prea larg. Verifică ce trebuie să fie scriibil și de ce. Nu transforma toată imaginea în 777 pentru a rezolva o permisiune de storage.

De evitat

Eticheta latest nu este un plan de reproducibilitate.

Atelier final: Containere cu date persistente

Timp de lucru estimat: 40 de minute. Folosește datele de laborator de mai jos și salvează rezultatul separat de orice proiect real.

Date de pornire

Mediu demonstrativ cu aplicație și bază de date, rețea internă și volum numit. Creează o înregistrare fictivă înainte de recrearea containerului aplicației.

1. Pregătire — 5 minute

Extrage intrările și condițiile din enunț. Notează ce rezultat trebuie produs și ce informații nu sunt confirmate. Pregătește fișierele sau tabelul în care vei păstra dovada lucrului.

2. Construire — 15 minute

Scrie Dockerfile și compose.yaml, explică fiecare mount și port, apoi recrează aplicația. Verifică datele și conectivitatea fără a publica inutil portul bazei.

3. Variație și verificare — 15 minute

Oprește temporar baza în laborator și urmărește eroarea aplicației. Documentează diferența dintre restart și ștergerea unui volum.

4. Autoevaluare — 5 minute

Compară rezultatul cu reperele de mai jos. Înregistrează cel puțin o eroare identificată sau un caz-limită verificat, plus corectura ori explicația comportamentului observat.

Repere de rezolvare

Datele persistă în volum. Numele serviciului permite comunicarea internă. Parola nu este inclusă în imagine sau în repository. Pornirea procesului nu este singura dovadă că baza este pregătită.

Ce predai

- Dockerfile
- compose.yaml
- Volum
- Health check
- README

Criterii de acceptare

- Baza nu este expusă inutil
- Containerul rulează fără root unde este posibil
- Secretele nu intră în imagine

Păstrează o versiune inițială și una revizuită, cu explicația diferențelor. Dacă un criteriu nu este îndeplinit, revino la lecția relevantă și repetă verificarea. Durata este orientativă; instalarea mediului și aprofundarea pot necesita timp suplimentar.

Registrul de lucru

Pentru fiecare etapă notează intrarea, rezultatul, dovada verificării, eroarea găsită și corectura. Nu introduce date personale sau secrete în fișa partajată.

Documentație și surse

- Get started with Docker — Docker

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