PLANOVI OBNOVE WEB POSLUŽITELJA

Švenjak, Roko

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Specijalistički diplomski stručni studij Informacijske tehnologije

ROKO ŠVENJAK

Z A V R Š N I R A D

PLANOVI OBNOVE WEB POSLUŽITELJA

Split, rujan 2022.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Specijalistički diplomski stručni studij Informacijske tehnologije

Predmet: Upravljanje poslužiteljima otvorenog kôda

Z A V R Š N I R A D

Kandidat: Roko Švenjak

Naslov rada: Planovi obnove web poslužitelja

Mentor: dipl.ing. Valentini Kožica, predavač

Split, rujan 2022.

Sadržaj

Sažetak

Cilj ovog rada je opisati instalaciju web aplikacije (Prestashop web prodavaonice) na Debian operacijskom sustavu kroz tri različita modela. Osim instalacije, također se prikazuje i obnova poslužitelja preko izrađene sigurnosne kopije za svaki model.

U prvom modelu se vrši instalacija web aplikacije na fizičkom (engl. *Bare-metal*) poslužitelju (Debian). U drugom modelu se vrši instalacija web aplikacije na virtualizatoru (Debian + virtualizator). Naposljetku, u trećem modelu se vrši instalacija web aplikacije na Docker platformi (Debian + Docker).

U radu su objašnjene razlike između svakog modela te su uspoređeni različiti parametri kao što su: kompleksnost instalacije, vrijeme izrade sigurnosne kopije, kompleksnost izrade sigurnosne kopije.

Ključne riječi: Sigurnosna kopija, oporavak, Debian, Virtualizacija, Docker

Summary

Web server recovery plans

The aim of this paper is to describe the installation of a web application (Prestashop web store) on a Debian operating system through three different models. In addition to the installation, the paper shows how to restore the server via the backup created for each model.

In the first model, the web application is installed on a bare-metal server (Debian). In the second model, the web application is installed on a virtualizer (Debian + virtualizer). Finally, in the third model, the web application is installed on the Docker platform (Debian + Docker).

The paper also explains the differences between each model and compares various parameters such as: complexity of the installation, backup time, backup complexity, and the like.

Keywords: Backup, Recovery, Debian, Virtualization, Docker

1. Uvod

U današnje doba podaci i informacije su kritične stavke za bilo koje poslovanje. U različite sustave se svakodnevno zapisuju raznoliki podaci. Primjerice podaci o kupcu, transakcijama, materijalima, računima, narudžbama i slično. Gubitak takvih podataka za poslovanje može biti dovoljno štetno da uzrokuje propast poslovanja.

Stoga da bi se smanjio ili potpuno uklonio rizik od gubitka podataka i informacija izrađuje se sigurnosna kopija. Međutim, danas postoji mnogo različitih sustava na kojima se može izvoditi aplikacija poslovanja. Kao primjer takvog sustava se može uzeti Prestashop aplikacija. Prestashop aplikacija je web program koja ima svrhu web prodavaonice. U ovom radu je specifično opisana izrada sigurnosne kopije i obnova Prestashop aplikacije. Aplikacija je opisana kroz tri modela rada:

1. Prestashop aplikacija na fizičkom poslužitelju
2. Prestashop aplikacija na virtualnom poslužitelju
3. Prestashop aplikacija na Docker platformi

U poglavlju „Tehnologije“ su opisani alati, okruženja i tehnologije potrebne za izradu i izvođenje Prestashop aplikacije te izradu njene sigurnosne kopije i vršenje oporavka.

U poglavlju „Fizički poslužitelj“ je detaljno opisan proces postavljanja okruženja Debian operacijskog sustava za rad i instalaciju Prestashop aplikacije. Također je prikazana izrada sigurnosne kopije i obnova poslužitelja i same aplikacije. Ponovno je isti proces opisan za poglavlja „Virtualni stroj“ i „Docker“ ali za različite modele. U poglavlju „Usporedba“ su uspoređeni parametri različitih modela.

Na samome kraju navodi se zaključak, korištena literatura te popis oznaka i referenci.

2. Tehnologije

U ovom poglavlju predstavljeni su alati i tehnologije potrebni za instalaciju i postavljanje okruženja poslužitelja i Prestashop aplikacije. Korišteni su softveri otvorenog kôda: Debian operacijski sustav, Apache, PHP, MySQL, QEMU virtualizator, KVM hipervizor, Docker, cron i RAID.

2.1. Debian operacijski sustav

Debian je Linux distribucija sastavljena od besplatnog softvera otvorenog kôda koja je razvijena od strane Debian Project kojoj pomaže zajednica [1]. Debian sustavi trenutno mogu koristiti Linux ili FreeBSD jezgru. Debian dolazi sa 59000 dostupnih paketa te APT (*Advanced Package Tool*) alatom pomoću kojeg traži, upravlja i šalje upite o paketima. APT alat dolazi sa sučeljem naredbenog retka visoke razine za olakšano korištenje. Također, Debian nudi CD i DVD datoteke koje sadržavaju različita grafička korisnička sučelja stoga korisnik može birati koji želi koristiti.

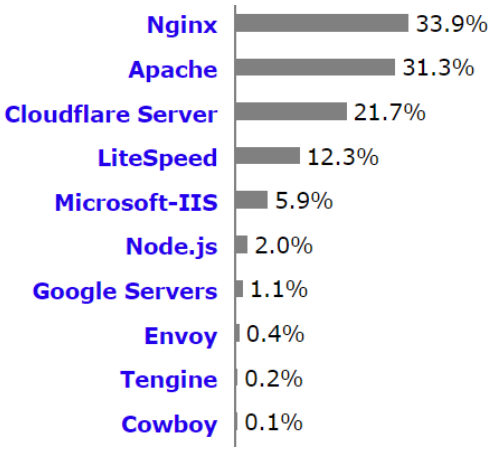
Debian je jedan od najpopularnijih izbora za operacijski sustav na web poslužiteljima. Popularan je jer je besplatan i planira ostati takav. Također se redovito razvija novi softver i sigurnosne zakrpe te je vrlo pouzdan i stabilan. Jako često se koristi kao temelj za korištenje LAMP (Linux, Apache, MySQL, PHP) tehnologija.

2.2. Apache

Apache je softver web poslužitelja koji je odgovoran za prihvaćanje HTTP (Hypertext Transfer Protocol) zahtjeva od posjetitelja i slanje natrag odgovora u obliku web stranica [2].

Apache djeluje kao posrednik između samog poslužitelja i klijentskog računala. Apache ima posao da poslužuje web stranice, tj. povlači sadržaj sa poslužitelja na svaki zahtjev korisnika i isporučuje ga korisniku.

Prva inačica softvera je puštena 1995. godine i bazirana je na NCSA HTTPd poslužitelju. Do 2021. godine Apache je bio najčešće korišten softver web poslužitelja i izvodio se na 35% svih web stranica na svijetu. Prošle godine ga je po popularnosti pretekao NGINX (Slika 1).



Slika 1: Postotci korištenosti softvera web poslužitelja [3]

2.3. PHP

PHP (rekurzivni akronim za *PHP: Hypertext Preprocessor*) je skriptni jezik otvorenog kôda i opće namjene koji je posebno prikladan za razvoj web stranica i može se ugraditi (engl. *Embedd*) u HTML dokument [4]. Za razliku od JavaScript jezika koji se izvršava na klijentskom računalu, PHP kôd se izvršava na poslužitelju, tj. generira se HTML dokument koji se šalje nazad klijentu.

PHP je dobro dokumentiran, održavan, brz i siguran te radi na više platformi (Windows, MacOS, Linux). Stoga je vrlo popularan izbor za skriptni jezik koji se izvršava na strani poslužitelja. Ima ugrađenu potporu za rad sa MySQL sustavom baze podataka, ali i podržava ostale sustave.

Prva inačica softvera je puštena 1995. godine, a danas se najčešće koristi verzija 7 (71.6% svih web stranica koje koriste PHP) iako je PHP verzija 8 puštena 2020. godine.

2.4. MySQL

MySQL je sustav otvorenog kôda za upravljanje relacijskim bazama podataka [5]. Oracle razvija, distribuira i podržava MySQL softver. Relacijska baza podataka je baza koja svoje podatke organizira u jednu ili više tablica i ti podaci mogu povezani (engl. *Relation*) međusobno. Takve veze pomažu pri organizaciji i rukovođenju podataka te sprečavaju stvaranje redundantnosti podataka.

Pomoću SQL (*Structured Query Language*) jezika se mogu izvršavati razne operacije nad tim podacima poput kreiranja, dohvaćanja i modificiranja. SQL je najčešće korišten standardizirani jezik koji se koristi za pristup bazama podataka.

Pomoću MySQL sustava podaci se trajno zapisuju na disk u bazu podataka na *host* poslužitelju. Kako bi se spriječio gubitak podataka zbog kvarova na poslužitelju mogu se izraditi sigurnosne kopije podataka.

Izrada sigurnosne kopije se može izvršavati dok je baza podataka spuštena (engl. *Offline*) ili podignuta (engl. *Online)*. Stoga se mogu razlikovati dva tipa izrade sigurnosne kopije:

- Hladna (engl. *Cold* ili *Offline*) izrada sigurnosne kopije
- Topla (engl. *Hot* ili *Online*) izrada sigurnosne kopije

Činjenicom da Prestashop obavlja funkciju web prodavaonice, poželjno je da je dostupna korisnicima što je više moguće. Stoga će se za ovaj rad prikazati izrada tople sigurnosne kopije za sva tri modela.

Izrada tople sigurnosne kopije predstavlja izazove koji se trebaju riješiti. Jedan od tih izazova je dosljednost (engl. *Consistency*) podataka u posljedičnoj sigurnosnoj kopiji. Izrada hladne sigurnosne kopije nema taj problem jer sustav ne zapisuje nove podatke dok se vrši kopija stoga će krajnja kopija biti dosljedna za sve podatke. Naravno, velika mana tog pristupa je što baza podataka nije dostupna dok je ugašena, a i samim time podaci unutar nje.

Za razliku od hladne, izrada tople sigurnosne kopije se događa dok se zapisuju novi podaci u bazu podataka stoga je nekako potrebno održati dosljednost posljedične kopije. Imajući to na umu može se birati između dva tipa tople izrade sigurnosne kopije:

- Fizička izrada sigurnosne kopije
- Logička izrada sigurnosne kopije

Fizička izrada sigurnosne kopije baze podataka je sirova kopija direktorija i datoteka koje pohranjuju sadržaj baze podataka [6]. Činjenicom da se mora paziti na dosljednost, nije moguće samo kopirati datoteke preko datotečnog sustava poslužitelja na drugu lokaciju. Takva kopija bi imala dio starih podataka i dio novih podataka koji su se u međuvremenu zapisali stoga takva kopija nije dosljedna.

MySQL nudi MySQL Enterprise proizvod koji adresira spomenuti problem dosljednosti prilikom fizičke izrade sigurnosne kopije. Proizvod nudi `mysqlbackup` alat, tj. naredbu koja izvršava proces fizičke izrade sigurnosne kopije pomoću MySQL servisa pazeći na dosljednost podataka putem raznih metoda. Ovaj proizvod je licenciran i nije besplatan.

Za razliku od fizičke, logička izrada sigurnosne kopije sprema informacije predstavljene kao logička struktura baze podataka i sadržaja [6]. MySQL također nudi alat `mysqldump` koji služi za izradu logičke sigurnosne kopije. Slično kao i `mysqlbackup`, `mysqldump` sadrži različite metode koje se brinu za dosljednost posljedične kopije dok se izvršava sama izrada kopije. No za razliku od `mysqlbackup`, `mysqldump` alat je potpuno besplatan i dolazi sa standardnom verzijom MySQL server paketa stoga će se koristiti u ovom radu za izradu sigurnosne kopije baze podataka.

Primjer naredbe za izradu sigurnosne kopije pomoću `mysqldump` alata izgleda ovako: `mysqldump --single-transaction --databases prestashop-db > full-backup.sql`. Posljedična datoteka `full-backup.sql` sadrži skup SQL naredbi koje se mogu preko `mysql` klijenta iskoristiti za oporavak zapisanih podataka i tablica.

Pomoću opcije `--single-transaction`, alat osigurava dosljednost kopije tako da esencijalno napravi snimku (*snapshot*) podataka prije početka izrađivanja kopije. Drugim riječima, promjene koje se dogode u InnoDB tablicama tijekom izrade kopije neće biti uključene u posljedičnu kopiju. Prestashop aplikacija koristi samo InnoDB tablice stoga je takva kopija prikladna za nju. Također, osim tijekom jako kratkog zaključavanja tablica (manje od 1 sekunde, ovisno o računalu) pri početku izvršavanja naredbe, na tablice baze podataka će se najnormalnije moći izvršavati operacije pisanja i čitanja što znači da će aplikacija nastaviti raditi bez problema.

Osim izrade kompletne sigurnosne kopije, pomoću `mysqldump` alata je moguća i inkrementalna sigurnosna kopija. Inkrementalna sigurnosna kopija je izrada kopije samo za podatke koji su se mijenjali od zadnje sigurnosne kopije. Obnova takve kopije je kompliciranija nego obnova kompletne kopije jer se mora obnoviti veći broj datoteka. Inkrementalna sigurnosna kopija je pogodna za baze podataka u koje se zapisuje velika

količina podataka. Prestashop baza podataka je malena i ne predviđa se zapisivanje ogromne količine podataka stoga se u radu opisuje izrada i obnova kompletne sigurnosne kopije.

2.5. QEMU i KVM

QEMU (*Quick Emulator*) je emulator i virtualizator otvorenog kôda [7]. Većinom se koristi kao virtualizator u kombinaciji sa komponentama KVM-a.

KVM (*Kernel-based Virtual Machine*) je virtualizacijska tehnologija otvorenog kôda koja je ugrađena u Linux jezgru (od 2007. godine). Pomoću KVM-a Linux obavlja funkciju hipervizora koji omogućuje poslužitelju pokretanje više izoliranih virtualnih okruženja [8]. Takvo virtualno okruženje se zove virtualni stroj.

Samostalno, QEMU je hipervizor drugog tipa i ne podržava izvornu (engl. *Native*) virtualizaciju i u takvoj konfiguraciji nema dobru izvedbu (engl. *Performance*). Kada hipervizor emulira procesor onda mora sve upute namijenjene emuliranom procesoru prevoditi fizičkom procesoru te će stoga izvedba sustava biti lošija.

Međutim, QEMU može raditi u kombinaciji sa KVM modulom i tada može izvoditi virtualne strojeve u izvornim brzinama izvođenja tako da kôd virtualnog stroja izvodi izravno na procesoru *host* poslužitelja. Danas većina procesora podržava takav način izvođenja, tj. podržavaju virtualizaciju. U takvoj konfiguraciji, QEMU postaje hipervizor prvog tipa sa puno boljom izvedbom.

Virtualni strojevi će se spremati u *image* datoteke posebnog formata `qcow2` (*QEMU copy-on-write 2*). Takve datoteke na disku *host* poslužitelja zauzimaju samo onoliko mjesta koliko je gostujući operacijski sustav zauzeo. Također, format omogućava kreiranje *overlay* datoteka koje zapisuju razliku od druge bazične *image* datoteke. Takav zapis datoteka omogućuje povratak stanja virtualnog stroja u određeni trenutak vremena i također može olakšati izradu sigurnosne kopije zbog nedopuštanja zapisivanja na datoteke.

2.6. Docker

Docker je otvorena platforma za razvoj, isporuku i pokretanje aplikacija [9]. Docker je mlada tehnologija (prva inačica puštena 2013. godine) koja je u aktivnom razviju. Za razliku od Linux operacijskih sustava, Windows i MacOS operacijski sustavi ne podržavaju izvorno Docker.

Pomoću Docker-a se mogu pakirati i pokretati aplikacije u izoliranim okruženjima zvanim spremnici (engl. *Container*). Zbog izolacije spremnika moguće je pokretati više spremnika na istom poslužitelju u isto vrijeme. Također, ti spremnici su „lagani“ i zauzimaju malo mjesta na disku. Najveća prednost spremnika je da oni u sebi sadržavaju sve što je potrebno da se izvodi aplikacija stoga korisnik ne mora brinuti koji alati se nalaze na operacijskom sustavu *host* poslužitelja.

Da bi kreirali i koristili Docker spremnik potrebno je imati Docker *image* datoteku. Ta datoteka je esencijalno predložak sa uputama za stvaranje Docker spremnika u kojoj će se aplikacija izvoditi. Jako često je jedna *image* datoteka bazirana na drugoj i dodatno će sadržavati neke prilagodbe.

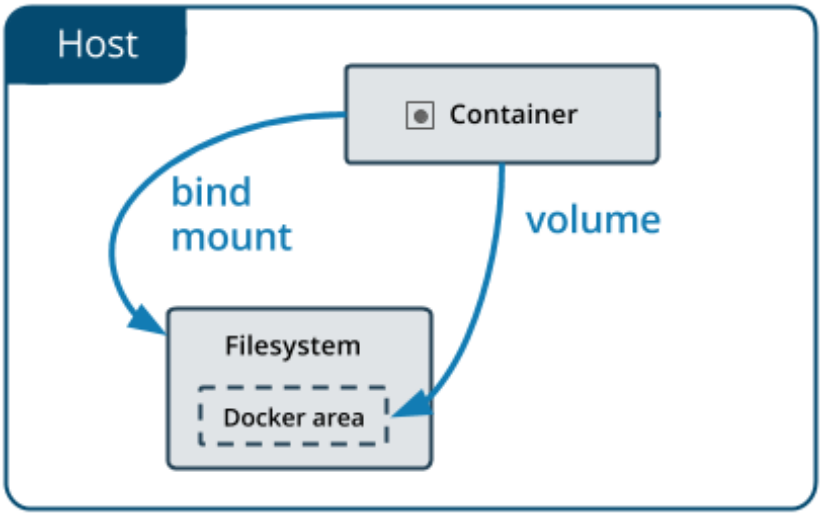
Korisnik može samostalno kreirati svoje Docker *image* datoteke ili može koristiti tuđe koje su objavljene u javnom registru. Da bi kreirao svoju *image* datoteku korisnik mora definirati Dockerfile datoteku koja sadrži instrukcije potrebne za kreiranje *image* datoteke. Svaka instrukcija u Dockerfile datoteci stvara sloj (engl. *Layer*) u *image* datoteci i ti slojevi se gledaju zasebno. Kad se *image* datoteka rekreira, ponovno se izgrađuju samo slojevi koji su se u međuvremenu mijenjali. Ovaj proces je ključan razlog zašto je proces izrade *image* datoteke brz i efikasan.

Korisnik također može koristiti i službene Docker *image* datoteke. Te datoteke se mogu pronaći na javnom repozitoriju zvanom Docker Hub. Takve Docker *image* datoteke su službene jer imaju određene značajke:

- Kreiraju ih službeni razvojni timovi
- Redovno se održavaju
- Drže se najboljih praksi pisanja Dockerfile datoteka
- Pružaju jasnu dokumentaciju za korištenje
- Osiguravaju da se sigurnosna ažuriranja primjenjuju na vrijeme
- Docker Inc. sponzorira posebni tim koji je odgovoran za pregled i objavljivanje svih sadržaja u službenim Docker *image* datotekama

Po standardnoj konfiguraciji Docker spremnici gube stanje i sve zapisane promjene nakon što se spremnici ugase i unište. Kako bi se podaci trajno zapisali Docker nudi dvije opcije: volumen ili *bind mount*.

Docker volumeni se pohranjuju u dijelu datotečnog sustava kojim upravlja Docker. Korisnik ne može izravno pristupati podacima unutar volumena preko datotečnog sustava poslužitelja već to mora odraditi putem Docker-a. Za razliku od volumena *bind mount* se može pohraniti bilo gdje na *host* poslužitelju (Slika 2).



Slika 2: Pohrana podataka spremnika

Docker volumeni su preporučeni tip trajne pohrane podataka spremnika na poslužitelju. Volumeni imaju neke prednosti nad *bind mount metodom* [10]:

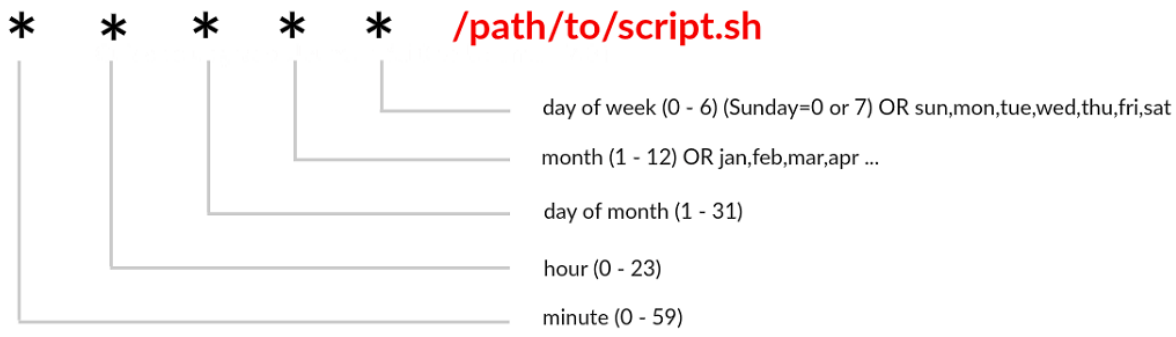
- Za volumene se lakše izrađuje sigurnosna kopija i lakše ih je migrirati.
- Može se upravljati volumenima preko Docker API-ja.
- Volumeni rade i na Linux i na Windows spremnicima.
- Volumeni se mogu sigurnije dijeliti između više spremnika.
- Spremnici mogu unaprijed popuniti sadržaj volumena.
- Imaju puno bolje izvedbe na Docker Desktop aplikaciji (aplikacija sa grafičkim sučeljem za rukovođenje spremnika).

2.7. Cron

Cron je softver koji korisniku omogućuje unos naredbi za izvršavanje zadataka u određeno vrijeme na UNIX sustavima. Zadaci koji se definiraju preko cron alata se zovu cron poslovi (engl. *Job*). Cron je *daemon*, tj. pozadinski proces koji izvršava poslove koji nisu interaktivni [11].

Stoga se cron može koristiti za automatizaciju izrade sigurnosne kopije preko definiranih poslova koji će se izvršavati u redovnim intervalima.

Da bi se napravio novi cron posao potrebno ga je izraditi u `crontab` konfiguracijskoj datoteci. Datoteci se može pristupiti preko naredbe `crontab -e`. Unutar datoteke se zapiše novi posao prema specifičnom formatu (Slika 3).



Slika 3: Crontab vremenski format

U ispisu (Ispis 1) se može vidjeti primjer cron posla unutar `crontab` datoteke koji će izvršavati skriptu `backup.sh` svaku nedjelju u 03:30.

```
30 3 * * SUN /put/do/backup.sh
```

Ispis 1: Primjer cron posla

2.8. RAID1 konfiguracija diskova

RAID (*Redundant Array of Independent Disks)* je način pohranjivanja istih podataka na različitim mjestima na više diskova radi zaštite podataka u slučaju kvara diska [12]. Postoji više RAID konfiguracija među kojima neke čak i ne koriste metodu redundantnosti podataka već im je cilj bolja izvedba.

RAID upravljač (engl. *Controller*) je uređaj koji služi za upravljanje diskovima unutar niza (engl. *Array*) diskova. RAID upravljač može biti hardverski ili softverski stoga RAID konfiguracija se može podijeliti na hardversku i softversku.

Hardverski RAID koristi fizički RAID upravljač koji upravlja nizom diskova. Upravljač može biti poseban uređaj kao cjelina koji ima jedinstvenu svrhu upravljanja RAID-om no najčešće je ugrađen u matičnu ploču poslužitelja.

Za razliku od hardverskog, softverski RAID ne koristi fizički upravljač već je upravljač program koji koristi resurse procesora i radne memorije. U suštini, obje konfiguracije rade isti posao.

U ovom radu će se specifično koristiti softverska RAID1 konfiguracija za redundantnost podataka. RAID1, poznat kao zrcaljenje diskova (*engl. Disk mirroring*), konfiguracija se sastoji od najmanje dva diska. Sadržaj jednog diska se duplicira (zrcali) na drugi disk tako da su im sadržaji identični. U slučaju kvara jednog diska drugi disk preuzima cjelokupni posao te poslužitelj nastavlja raditi kao da se ništa nije dogodilo. Na taj način podaci na disku ostaju zaštićeni preko redundancije.

Posljedica RAID1 konfiguracije za dva diska je da se posao pisanja na diskove udvostručuje jer se sve mora duplicirati, ali se zato poboljšavaju izvedbe čitanja sa diskova jer se oba diska mogu čitati u isto vrijeme. Također, kapacitet diskova se prepolovi jer se gubi pohrana za spremanje novih podataka na drugom disku.

3. Fizički poslužitelj

U poglavlju je opisano postavljanje okruženja te proces instalacije i rad Prestashop aplikacije na fizičkom (engl. *Bare-metal*) poslužitelju. Fizički poslužitelj je računalni poslužitelj kojeg koristi samo jedan potrošač [13]. Na poslužitelju je instaliran Debian 11 (*bullseye)* operacijski sustav. Naposljetku je objašnjeno kako izraditi sigurnosnu kopiju važnih datoteka poslužitelja i aplikacije te oporavak istih.

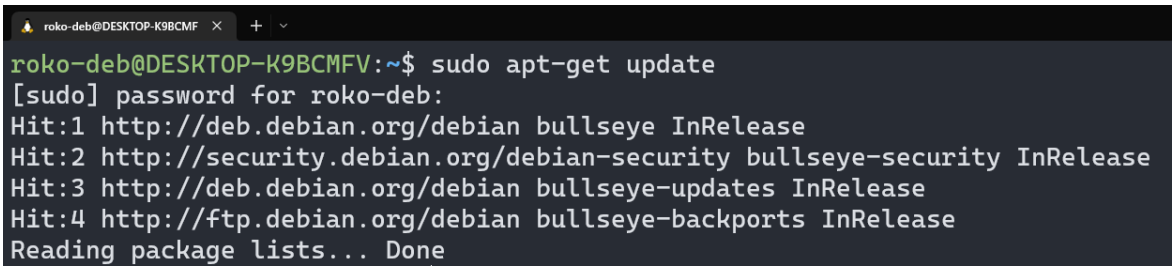
3.1. Postavljanje okruženja

Kao preduvjet za rad aplikacije potrebno je instalirati:

- Apache (verzije 2.2 ili iznad) – web poslužitelj
- MySQL (verzije 5.6 ili iznad) – sustav za pohranu i korištenje podataka
- PHP (verzije 7.4) – tumač jezika (engl. *Language interpreter*)
- PhpMyAdmin (opcionalan) – alat za administraciju nad MySQL-om

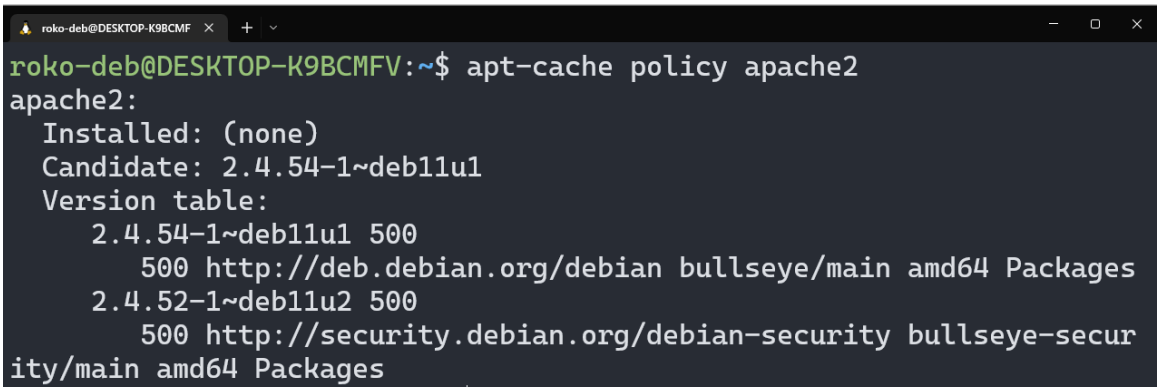
Unutar Debian 11 operacijskog sustava se može koristiti alat APT (*Advanced Package Tool*) za upravljanje paketima. Pomoću APT alata paketi će se preuzeti sa dostupnih repozitorija te nakon toga instalirati na poslužitelju.

Da bi se dostupni repozitoriji i njihov sadržaj ažurirao, prvenstveno je potrebno izvršiti naredbu `sudo apt-get update` unutar terminala. Na slici (Slika 4) se može vidjeti popis repozitorija koji je trenutno dostupan.



Slika 4: Ispis terminala za `sudo apt-get update` naredbu

Prije instalacije Apache softvera može se provjeriti koja je verzija dostupna u repozitorijima pomoću naredbe `apt-cache policy apache2`. Iz slike (Slika 5) se vidi da je dostupna verzija `2.4.54` unutar repozitorija `deb.debian.org/debian`.



Slika 5: APT detalji paketa Apache2

U trenutku pisanja rada verzija `2.4.54` Apache paketa je zadnja stabilna verzija koja zadovoljava uvjete Prestashop aplikacije stoga se može jednostavno instalirati preko naredbe `sudo apt-get install apache2`.

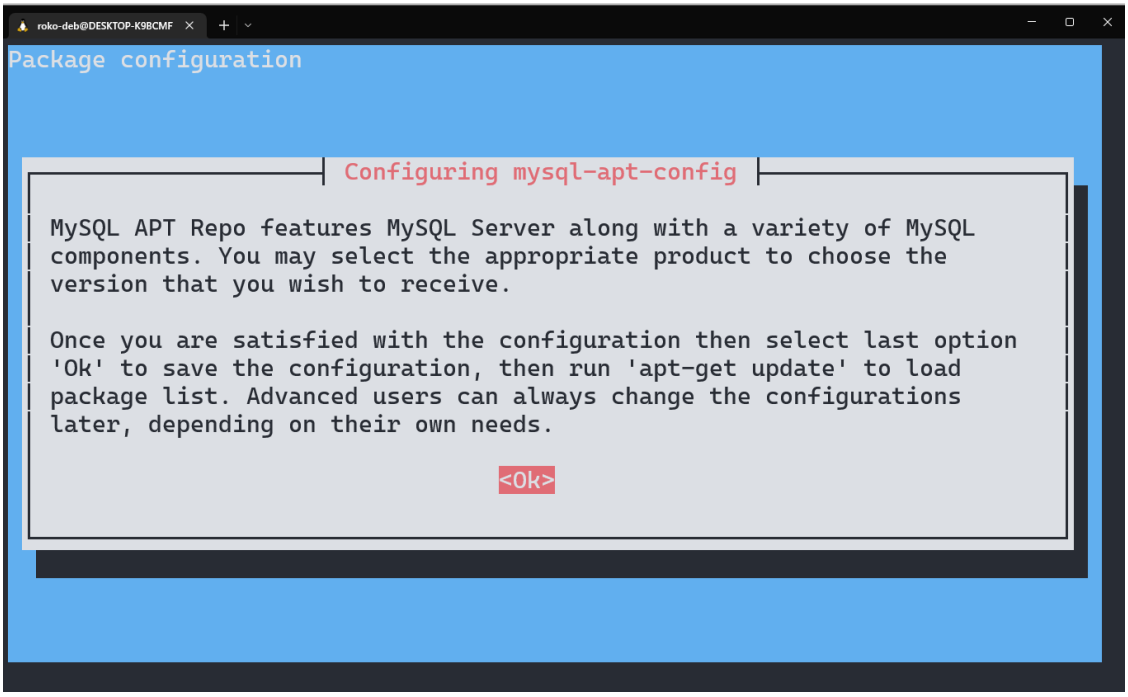
Po standardu, MySQL paket nije uključen u dostupnim repozitorijima i ne može se izravno instalirati. Da bi se preuzeo MySQL paket preko APT alata, potrebno je dodati MySQL APT repozitorij. Dodavanje repozitorija je moguće uraditi preko MySQL konfiguracijske datoteke koja se može preuzeti sa službene stranice. Za preuzimanje datoteke se može koristiti WGET alat preko HTTPS protokola. Za instalaciju MySQL alata verzije `8.0` koja je kompatibilna sa Prestashop aplikacijom potrebno je preuzeti konfiguracijsku datoteku verzije `0.8.22`.

Navedeni koraci se mogu odraditi preko naredbi u ispisu (Ispis 2):

```
sudo apt update && sudo apt -y  install wget gnupg lsb-release
wget https://repo.mysql.com//mysql-apt-config_0.8.22-1_all.deb
sudo dpkg -i mysql-apt-config_0.8.22-1_all.deb
```

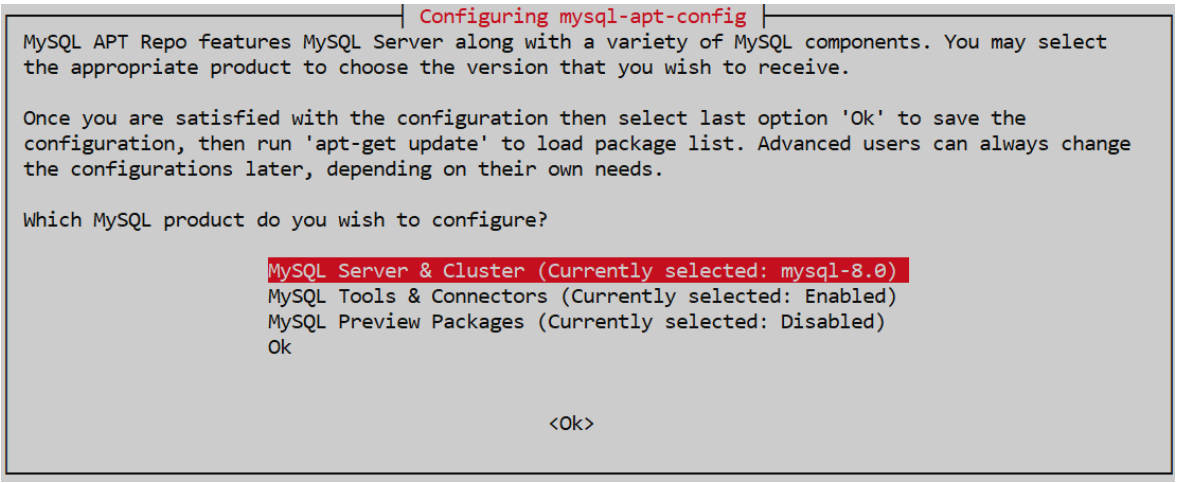
Ispis 2: Naredbe za instalaciju MySQL APT repozitorija

Nakon pokretanja svih naredbi treba se dobiti interaktivni zaslon (Slika 6).



Slika 6: Početni zaslon konfiguracije MySQL repozitorija

U ostatku instalacije potrebno je odabrati verziju alata koja se želi instalirati i sa time je konfiguracija gotova (Slika 7).



Slika 7: Zaslon za odabir verzije MySQL alata

Iza toga je potrebno opet izvršiti naredbu `sudo apt-get update` da bi se novi repozitorij ažurirao te je onda paket spreman za instalaciju. Instalacija paketa preko APT alata je sad moguća preko naredbe `sudo apt-get install mysql-server`. Tijekom instalacije paketa od korisnika se traži da unese lozinku za korijenskog (engl. *Root*) korisnika. Sa tim korakom proces instalacije MySQL alata je završen.

Kao i Apache, u trenutku pisanja rada PHP paket verzije `7.4` je moguće preuzeti iz inicijalno dostupnih repozitorija stoga je instalacija vrlo jednostavna. Paket se može instalirati preko naredbe `sudo apt-get install php7.4`.

Također, PhpMyAdmin alat se može preuzeti izravno sa dostupnih repozitorija preko naredbe `sudo apt-get install phpmyadmin`. Da bi alat ispravno radio potrebno je postaviti konfiguraciju pomoću naredbi u ispisu (Ispis 3):

```
sudo cp /etc/phpmyadmin/apache.conf /etc/apache2/conf-
available/phpmyadmin.conf
sudo a2enconf phpmyadmin
```

Ispis 3: Naredbe za konfiguriranje PhpMyAdmin alata

Alatu je moguće pristupiti na poveznici http://localhost/phpmyadmin/.

3.2. Konfiguracija i moduli

Osim same instalacije također je potrebno i konfigurirati alate te instalirati dodatne module za PHP i MySQL da bi aplikacija u potpunosti radila (Ispis 4). Ekstenzije i moduli, tj. paketi koji su dodatno potrebni se mogu instalirati preko dostupnih repozitorija (Slika 5).

```
sudo apt-get install libapache2-mod-php7.4 php7.4-common php7.4-curl
php7.4-gd php7.4-imagick php7.4-mbstring php7.4-mysql php7.4-json php7.4-
xsl php7.4-intl php7.4-zip php7.4-xml php7.4-fileinfo php7.4-iconv
php7.4-pdo php7.4-pdo-mysql php7.4-simplexml php7.4-bcmath php7.4-
memcached
```

Ispis 4: Naredba za instalaciju dodatnih modula i paketa

Nakon instalacije dodatnih paketa potrebno je omogućiti `mod_rewrite` modul pomoću naredbe `sudo a2enmod rewrite`. Nadalje potrebno je konfigurirati PHP konfiguracijsku datoteku `php.ini` na putanji `/etc/php/7.4/apache2/`. Unutar datoteke je potrebno promijeniti stavke na vrijednosti:

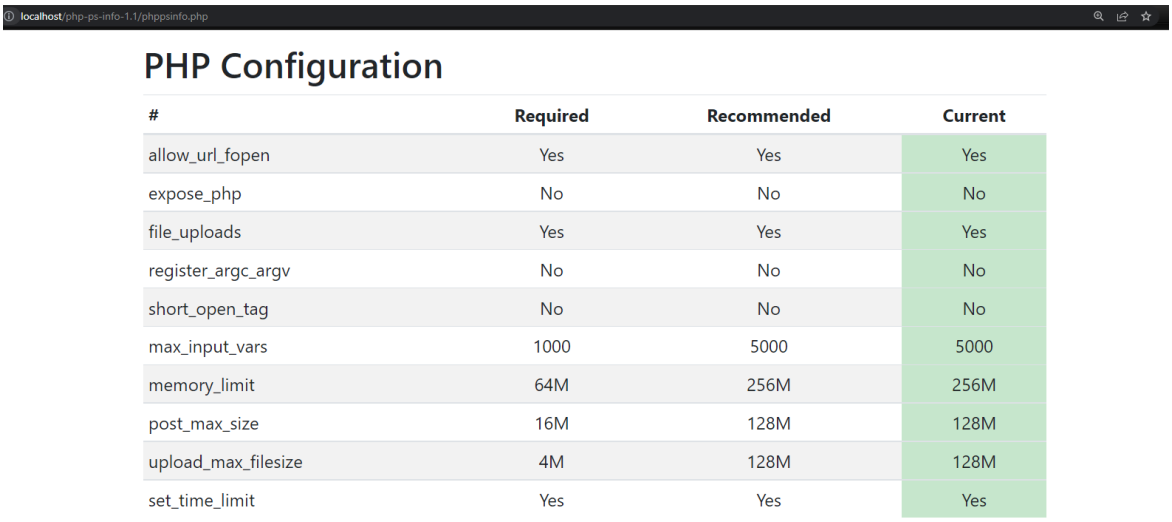
- `max_input_vars = 5000`
- `memory_limit = 256M`

- `post_max_size = 128M`
- `upload_max_filesize = 128M`

Također se treba pobrinuti da je postavka `allow_url_fopen` postavljena na vrijednost „`on`“. Ova postavka omogućuje Prestashop aplikaciji pristup udaljenim (engl. *Remote*) datotekama. Važno je omogućiti pristup jer je ključan dio prilikom procesa plaćanja. Nakon svih promjena potrebno je ponovno pokrenuti Apache servis pomoću naredbe `sudo service apache2 restart` da bi se promjene primijenile.

Kako bi provjerili da li su svi konfiguracijski parametri točno postavljeni može se koristiti `phppsinfo.php` datoteka koja se može preuzeti sa službenog Prestashop Github repozitorija (https://github.com/PrestaShop/php-ps-info/releases). Nakon preuzimanja datoteka se može raspakirati u direktorij `/var/www/html/` jer će se otvoriti pomoću web preglednika. Nakon raspakiranja, sadržaju datoteke se pristupa preko poveznice http://localhost/phppsinfo.php.

Otvaranjem poveznice se pojavljuje zaslon pomoću kojeg se utvrđuje da li je PHP konfiguracija ispravno postavljena (Slika 8).



Slika 8: PHP sistemski zahtjevi

Također, ako se zaslon pomakne prema dolje može se provjeriti i da li su sve PHP ekstenzije i moduli ispravno instalirani (Slika 9).

PHP Extensions

#	Required	Recommended	Current
BCMath Arbitrary Precision Mathematics	No	Yes	Yes
Client URL Library (Curl)	Yes	Yes	Yes
Image Processing and GD	Yes	Yes	Yes
Image Processing (ImageMagick)	No	Yes	Yes
Internationalization Functions (Intl)	Yes	Yes	Yes
Memcache	No	No	No
Memcached	No	Yes	Yes
Multibyte String (Mbstring)	Yes	Yes	Yes
OpenSSL	Yes	Yes	Yes
File Information (Fileinfo)	Yes	Yes	Yes
JavaScript Object Notation (Json)	Yes	Yes	Yes
PDO and MySQL Functions	Yes	Yes	Yes
PHP-DOM and PHP-XML	Yes	Yes	Yes
Zip	Yes	Yes	Yes

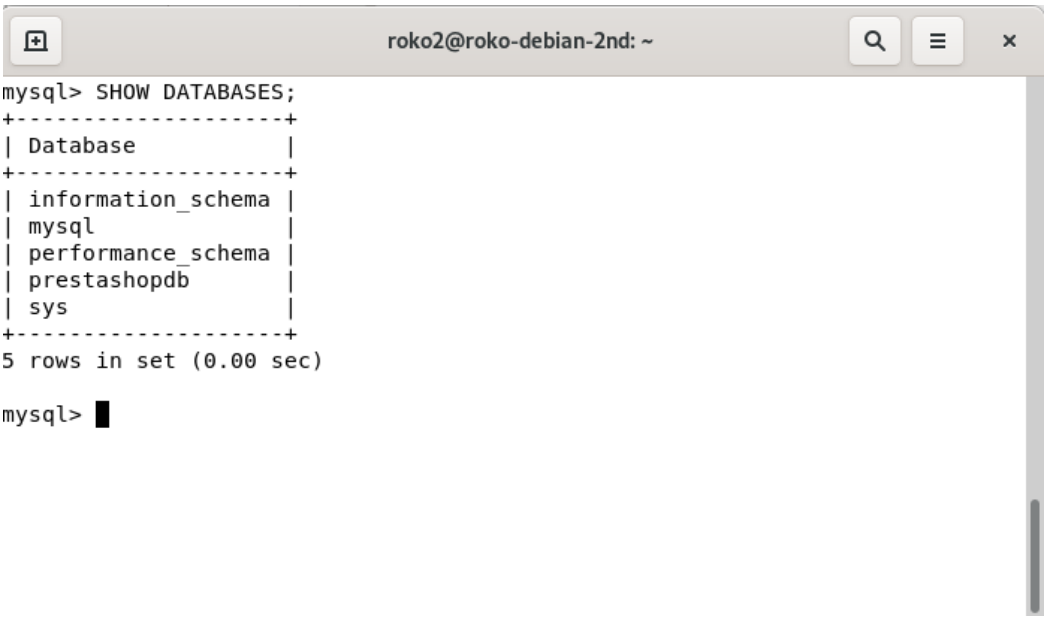
Slika 9: Status PHP ekstenzija

Iz Slika 8 i Slika 9 se može vidjeti da je sve ispravno konfigurirano i stoga je proces konfiguracije sa time završen.

3.3. Izrada MySQL baze podataka

Prije same instalacije Prestashop aplikacije preostalo je samo kreirati bazu podataka u kojoj će aplikacija spremati podatke. Kreiranje baze podataka se radi preko `mysql` klijenta naredbenog retka (engl. *Command-Line Client*). Preko naredbe `mysql -u root -p` se pristupa `mysql` klijentu i korisnik mora unijeti lozinku za pristup. Nakon toga bazu podataka je jednostavno moguće izraditi preko SQL naredbe `CREATE DATABASE prestashopdb COLLATE utf8mb4_general_ci` unutar klijenta.

Da bi potvrdili da li je `prestashopdb` baza podataka ispravno izrađena može se provjeriti preko naredbe `SHOW DATABASES` (Slika 10).

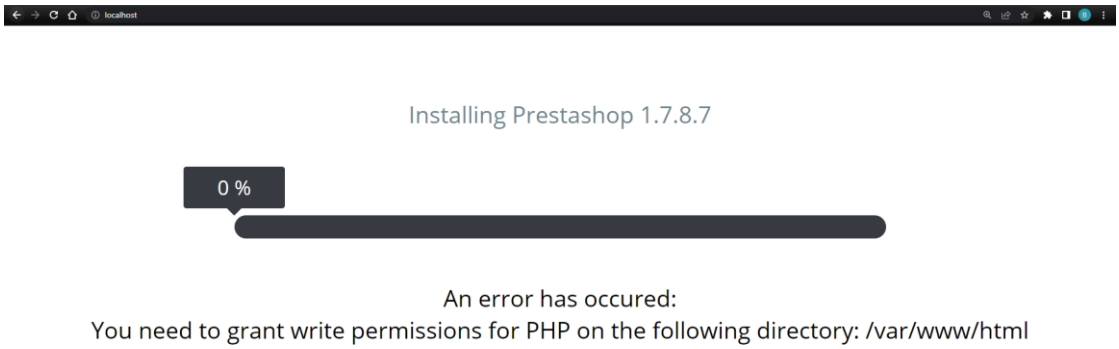


Slika 10: Rezultat naredbe za ispis postojećih MySQL baza podataka

3.4. Instalacija Prestashop aplikacije

Nakon preuzimanja potrebnih alata, postavljanja okruženja i konfiguracije te izrade baze podataka Prestashop aplikacija je spremna za instalaciju. Prvo, zadnju stabilnu verziju Prestashop-a potrebno je preuzeti sa službenog repozitorija pomoću `wget` alata i naredbe `wget https://download.prestashop.com/download/releases/prestashop_1.7.8.7.zip`.

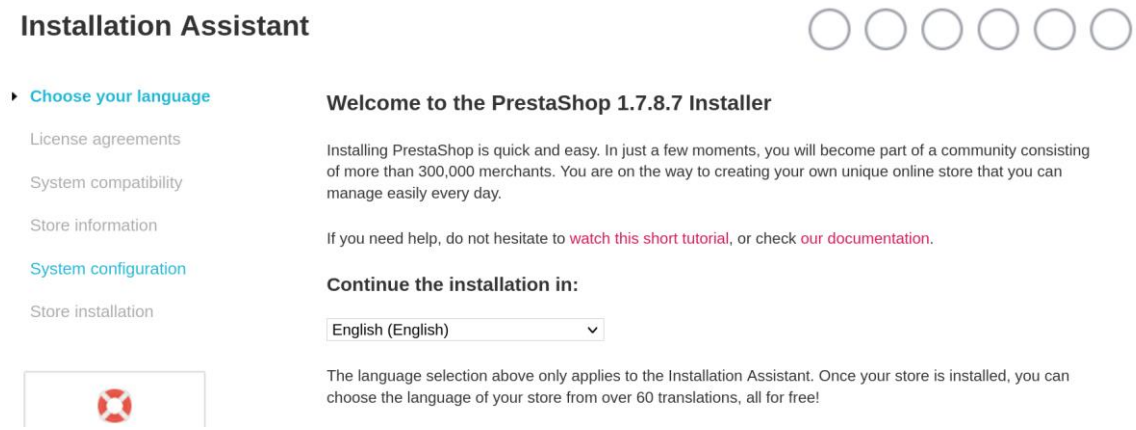
Preuzeta arhiva sadrži sve datoteke potrebne za instalaciju aplikacije te samu instalacijsku datoteku čije upute se moraju pratiti. Kako bi pristupili instalaciji aplikacije na poveznici `http://localhost/` arhiva se mora raspakirati u direktorij `/var/www/html`. Posjećivanjem poveznice se dobije zaslon na kojem piše da se dogodila greška (Slika 11).



Slika 11: Zaslon sa greškom

Greška se pojavljuje jer potrebni alati ne mogu pristupiti sadržaju unutar direktorija `/var/www/html`. Grešku je moguće otkloniti tako da se vlasništvo direktorija dodijeli grupi `www-data` jer Apache i PHP pristupaju datotekama koristeći tu grupu. Kada se dodijeli vlasništvo još je samo potrebno dati *write* prava nad direktorijem tako da se mogu zapisivati nove promjene unutar direktorija. To se radi pomoću naredbe `sudo chgrp -R www-data /var/www/html && sudo chmod g+w /var/www/html/`. Iza izvršenja naredbe se treba osvježiti prozor preglednika i sa time će se pokrenuti skripta za pokretanje instalacije Prestashop aplikacije.

Na zaslonu bi se trebao pojaviti pomoćnik za instalaciju (Slika 12).

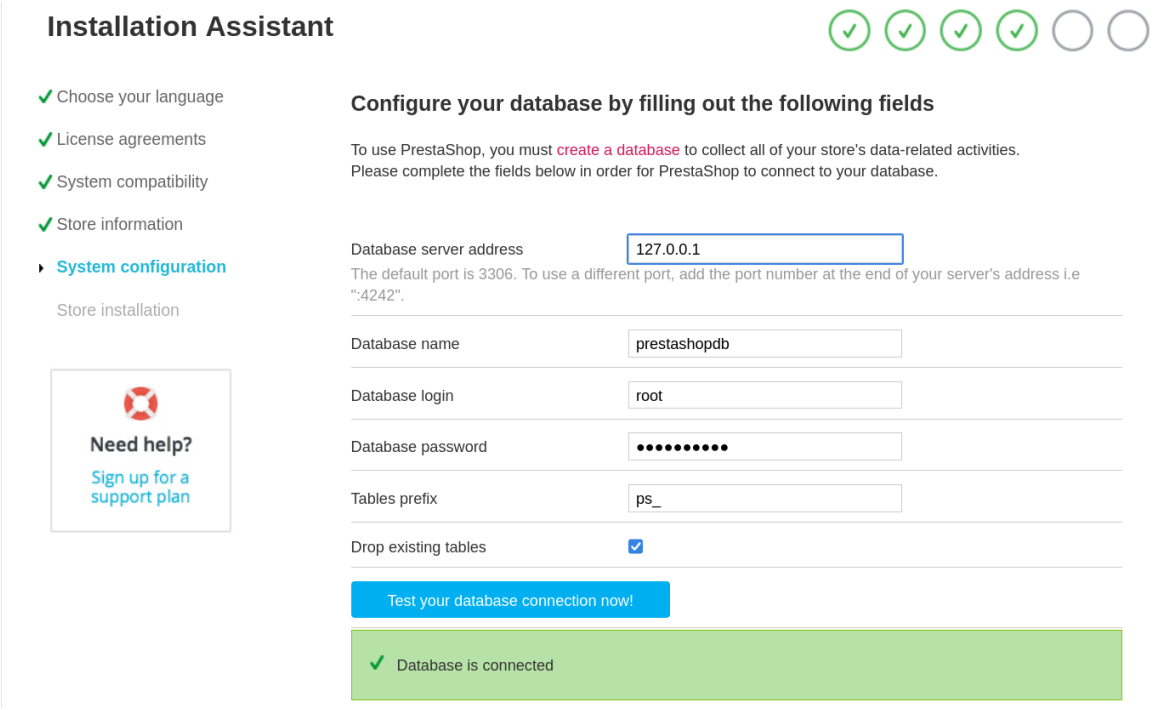


Slika 12: Zaslon sa pomoćnikom za instalaciju

U sljedećim prozorima je potrebno ispuniti informacije o prodavaonici te određene konfiguracijske parametre za sustav. Među bitnijim parametrima se ističu:

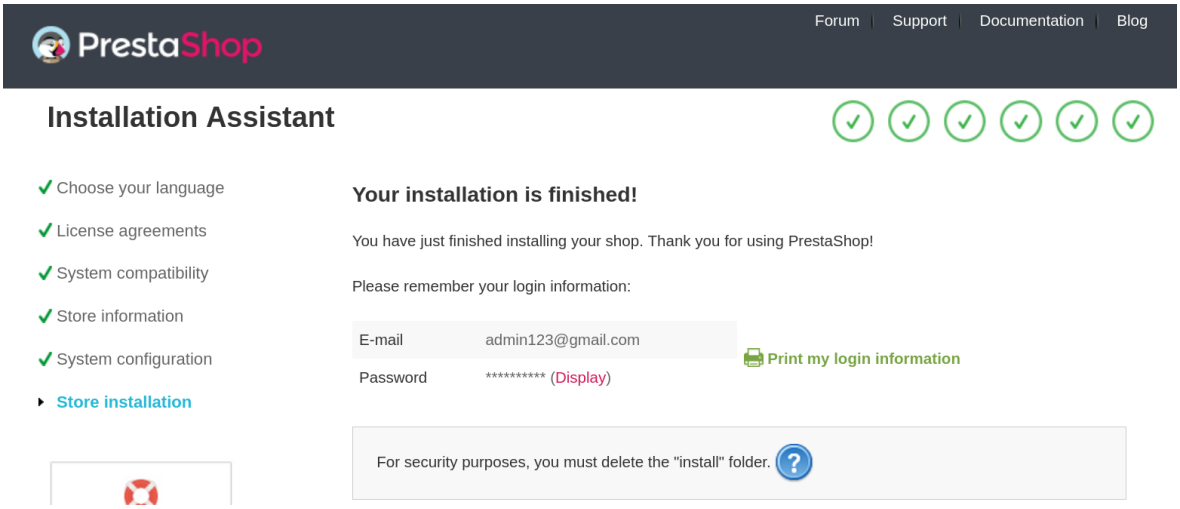
- *E-mail address* – koristi se za pristup administracijskom sučelju
- *Database server address* – adresa poslužitelja na kojoj se nalazi baza podataka (`127.0.0.1`)
- *Database name* – ime baze podataka u kojoj se spremaju podaci (`prestashopdb`)
- *Database login* – ime korisničkog računa preko kojega se spaja u MySQL sustav (`root`)
- *Database password* – lozinka koja se koristi za pristup MySQL sustavu

Nakon što su svi parametri baze podataka ispunjeni treba se pojaviti obavijest „*Database is connected*“ nakon pritiska na gumb „*Test your database connection now*“ (Slika 13).



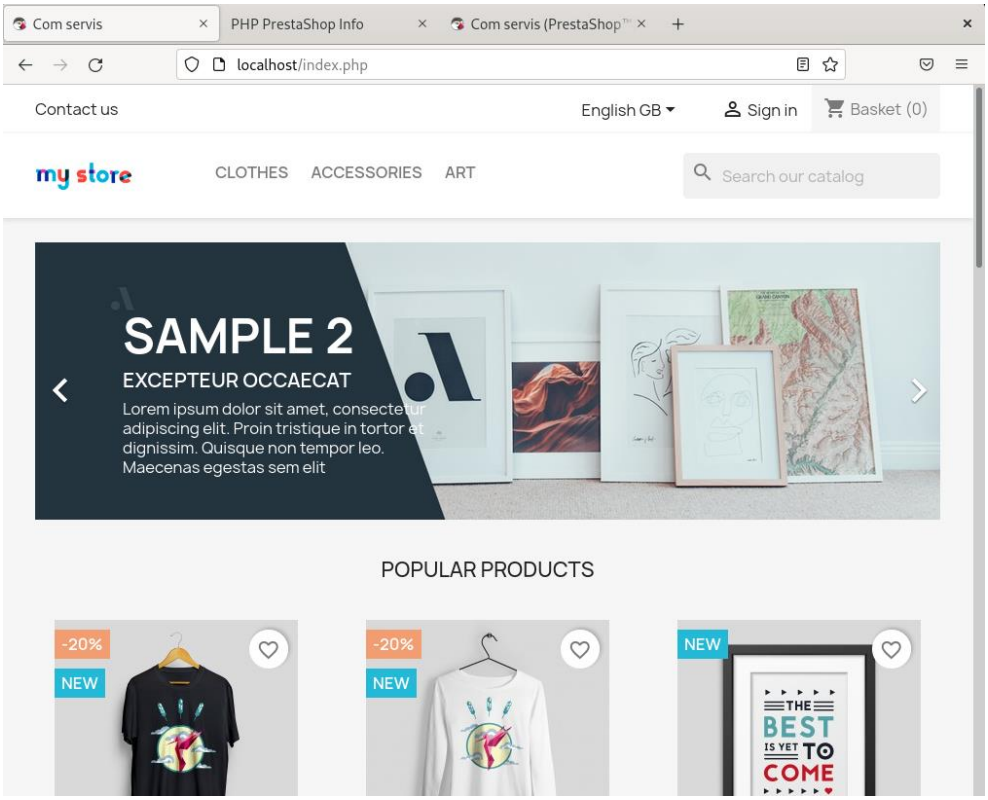
Slika 13: Zaslon sa postavkama baze podataka

Pritiskom gumba za sljedeću stranicu instalacija se dovršava te je aplikacija spremna za rad. Pomoćnik bi na zadnjem zaslonu trebao dati obavijest da je proces instalacije dovršen. (Slika 14).



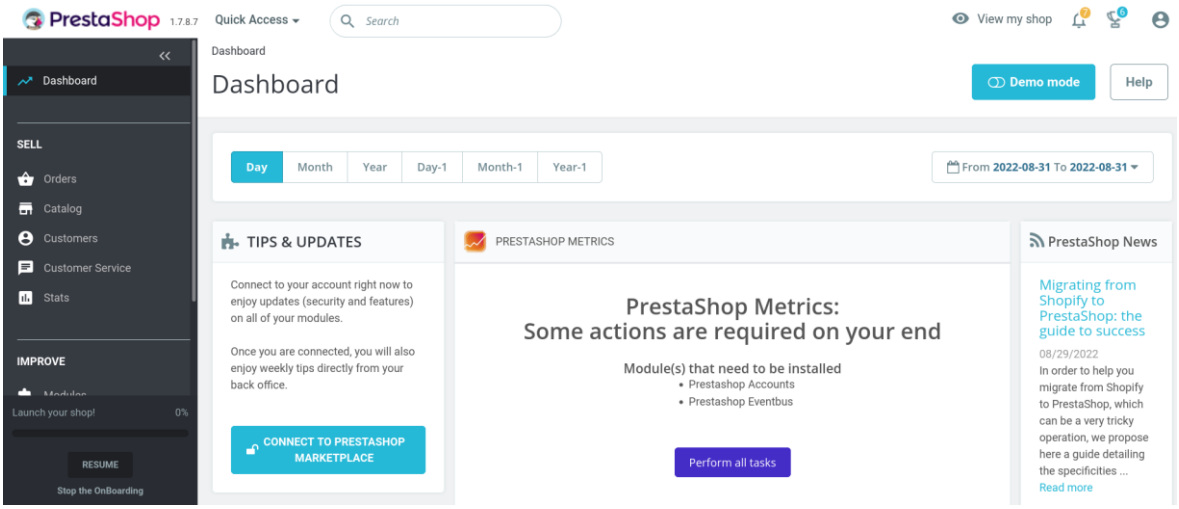
Slika 14: Zaslon završetka instalacije Prestashop aplikacije

Naposljetku, grafičkom sučelju aplikacije se može pristupiti na poveznici http://localhost/ (Slika 15).



Slika 15: Početni zaslon Prestashop prodavaonice

Administracijskom sučelju je moguće pristupiti preko poveznice http://localhost/admin (Slika 16). Prije odlaska na poveznicu, potrebno je izbrisati `install` direktorij unutar `/var/www/html` direktorija zbog sigurnosnih razloga.



Slika 16: Početni zaslon administracijskog sučelja

3.5. Izrada sigurnosne kopije

Sigurnosna kopija bi trebala sadržavati sve važne podatke i datoteke poslužitelja i Prestashop aplikacije pomoću koje bi se izvršila obnova poslužitelja u slučaju gubitka podataka ili kvara poslužitelja. Prestashop verzije `1.7.8.7` zahtijeva alate i okruženja specifičnih verzija te u suprotnome neće ispravno raditi. Stoga je potrebno napraviti sigurnosnu kopiju točno tih verzija alata i paketa kako bi se aplikacija ispravno obnovila.

Bitni segmenti i podaci koji su potrebni za obnovu Prestashop aplikacije su:

- Verzija operacijskog sustava i jezgre (engl. *Kernel*)
- Lista instaliranih paketa na operacijskom sustavu
- Dodani APT repozitoriji
- Konfiguracijske datoteke
- Datoteke Prestashop web aplikacije
- Podaci pohranjeni u bazi podataka

Izrada sigurnosne kopije navedenih podataka je moguća preko skripte `complete-backup.sh` (Ispis 5).

```
#! /bin/bash
CURRENT_DATE=$(date +"%d_%m_%Y")
BACKUP_DIR=backups/$CURRENT_DATE

sudo mkdir -p /$BACKUP_DIR
sudo mkdir -p /$BACKUP_DIR/configs
sudo mkdir -p /$BACKUP_DIR/prestashop-app-files
sudo touch /$BACKUP_DIR/server-info
sudo chmod 666 /$BACKUP_DIR/server-info
sudo touch /$BACKUP_DIR/packages_list_backup
sudo chmod 666 /$BACKUP_DIR/packages_list_backup
sudo touch /$BACKUP_DIR/full-backup-$CURRENT_DATE.sql
sudo chmod 666 /$BACKUP_DIR/full-backup-$CURRENT_DATE.sql

# 1) Verzija operacijskog sustava i jezgre
sudo cat /etc/debian_version > /$BACKUP_DIR/server-info
sudo uname -rv >> /$BACKUP_DIR/server-info
# 2) Debian instalirani paketi i dodatni APT repozitoriji
```

```
sudo dpkg -l | grep ^ii | awk '{print $2}' >
/$BACKUP_DIR/packages_list_backup
# 3) Konfiguracijske datoteke
sudo rsync -av /etc/apache2 /$BACKUP_DIR/configs/
sudo rsync -av /etc/mysql /$BACKUP_DIR/configs/
sudo rsync -av /etc/php /$BACKUP_DIR/configs/
sudo rsync -av /etc/phpmyadmin /$BACKUP_DIR/configs/
# 4) Prestashop datoteke
sudo rsync -av /var/www/html/ /$BACKUP_DIR/prestashop-app-files
# 5) Sigurnosna kopija baze podataka
sudo mysqldump --single-transaction --databases prestashopdb >
/$BACKUP_DIR/full-backup-$CURRENT_DATE.sql
# 6) Kopiraj sigurnosnu kopiju na sigurnu lokaciju
sudo rsync -av /$BACKUP_DIR/ /media/$USER/external_drive/
```

Ispis 5: Skripta za izradu sigurnosne kopije fizičkog poslužitelja

Prije izvršenja skripte je potrebno instalirati rsync (*remote sync*) alat ako prethodno nije instaliran na poslužitelju. Rsync je brz i svestran alat za kopiranje datoteka. Poznat je po svom algoritmu delta-prijenosa koji smanjuje količinu kopiranih podataka kopirajući samo razliku između izvornih datoteka i postojećih datoteka na odredištu [14]. Također je vrlo efikasan zbog mogućnosti očuvanja prava nad datotekama i direktorijima prilikom kopiranja. Rsync omogućava i prijenos podataka putem SSH (*Secure Shell*) protokola stoga se mogu slati podaci na sigurnu vanjsku lokaciju preko mreže.

Nakon izvršenja skripte na poslužitelju se može pronaći direktorij `backups` koji sadrži sigurnosnu kopiju. Posao koji izvršava kôd na ispisu (Ispis 5) se može podijeliti u više koraka. Prvo, definiraju se varijable za veću čitkost i preglednost kôda unutar skripte. Nakon toga izrađuju se direktoriji i datoteke u koje će se spremiti sigurnosna kopija. Naravno, potrebno je dati i potrebna prava tako da se mogu zapisivati promjene.

U prvom koraku počinje izrada sigurnosne kopije. Pomoću naredbi `cat` i `uname` izvlače se potrebne informacije o poslužitelju i jezgri te se spremaju u tekstualnu datoteku `server-info`.

U drugom koraku pomoću `dpkg -l` dohvaćaju se svi paketi koji se nalaze na poslužitelju. Nadalje sa `grep ^ii` se filtriraju oni paketi koji su statusa „*installed*“ i sa

`awk '{print $2}'` ponovno se filtrira dobiveni tekst tako da se dohvate samo imena paketa. Rezultat naredbe je popis svih instaliranih paketa na operacijskom sustavu i sprema se u tekstualnu datoteku `packages_list_backup`.

U trećem koraku se kopiraju sve važne konfiguracijske datoteke za Apache, MySQL, PHP i PhpMyAdmin alate. Te datoteke su bitne za obnovu jer je potrebno vratiti postavke alata specifične za rad Prestashop aplikacije. U četvrtom koraku se kopiraju datoteke same aplikacije.

U petom koraku se vrši logička vruća sigurnosna kopija baze podataka. Niz SQL naredbi se sprema u *dump* (`.sql`) datoteku koji služi za obnovu `prestashopdb` bazu podataka. Umjesto da se `mysqldump` naredbi izravno pruži korisničko ime i lozinka unutar naredbe, moguće je napraviti tekstualnu datoteku `/etc/.my.cnf` (Ispis 6) gdje se mogu zapisati lozinka i korisničko ime MySQL korisnika. Tu datoteku će `mysql` servis čitati kao globalnu konfiguracijsku datoteku i iščitati će vjerodajnice iz te datoteke prilikom pokretanja `mysql` klijenta. Sa time se postiže dodatna sigurnost vjerodajnica korisnika.

```
[client]
user=root
password=lozinka
```

Ispis 6: Primjer sadržaja datoteke `.my.cnf`

U šestom koraku se vrši kopija cijelog `backups` direktorija na neku sigurnu lokaciju. U ovom slučaju sigurnosna kopija se kopira na vanjski tvrdi disk koji je spojen na poslužitelj. Sa time se postiže da se podaci očuvaju u slučaju kvara poslužitelja i nemogućnosti pristupa podacima.

Vanjsko odredište kopije može biti i neko drugo proizvoljno odredište. Primjerice, odredište kopije može biti drugi poslužitelj koji je na drugoj geografskoj lokaciji spojen putem mreže na prvi poslužitelj. Sa takvim pristupom se osigurava dodatna sigurnost podataka jer u slučaju vremenske nepogode i uništenja originalnog poslužitelja sigurnosna kopija je i dalje očuvana na drugoj sigurnoj lokaciji.

3.5.1 Automatizacija pomoću cron alata

Ručno pokretanje skripte za izradu sigurnosne kopije svaki dan nije efikasno utrošeno vrijeme. Umjesto ručnog rada preporuča se raditi automatska izrada sigurnosne kopije pomoću cron alata.

Prethodno je opisana skripta (Ispis 5) za izradu sigurnosne kopije. Automatizaciju je jednostavno podesiti tako da se skripta izvršava u redovnim intervalima. Za Prestashop aplikaciju dovoljno je raditi sigurnosnu kopiju jednom dnevno u 03:00 ujutro. Stoga će se dodati cron posao u `crontab` konfiguracijsku datoteku (Ispis 7):

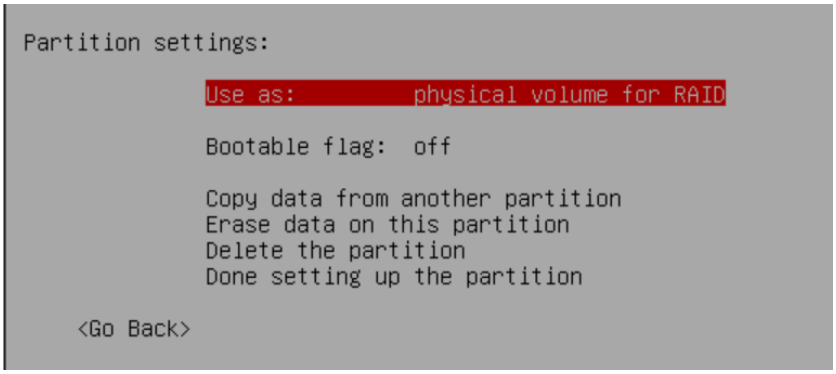
```
0 3 * * * /backup-scripts/complete-backup.sh
```

Ispis 7: Cron posao

3.5.2 RAID1 zrcaljenje

U poglavlju 2.8 je objašnjeno što je RAID1 i zašto se koristi za dodatnu sigurnost podataka. Kako bi se omogućila redundancija podataka i sa time smanjio rizik gubitaka podataka u slučaju kvara moguće je postaviti softverski RAID1 prilikom instalacije operacijskog sustava. Za izradu RAID1 konfiguracije potrebno je minimalno dva diska.

RAID1 konfiguraciju diskova je moguće postaviti tijekom instalacije operacijskog sustava. Tijekom instalacije operacijskog sustava kad se dođe na zaslon „*Partition disks*“ potrebno je odabrati ručno particioniranje. Nakon toga za oba diska je potrebno kreirati particije i postaviti da se koristi kao fizički volumen za RAID (Slika 17).



Slika 17: Postavke particije za RAID

U sljedećem zaslonu sa izbornika se odabere konfiguriranje softverskog RAID-a te u konfiguraciji za vrstu je potrebno postaviti na RAID1. Naposljetku, treba se upisati da se koriste dva aktivna uređaja u RAID polju i nužno je označiti particije koje će se koristiti za to RAID polje.

3.6. Obnova poslužitelja i Prestashop aplikacije

Ako je proces izrade sigurnosne kopije izvršen bez greški obnova poslužitelja će biti bezbolna. Osim očite prednosti očuvanja bitnih podataka klijenata i poslovanja, sigurnosna kopija također pojednostavljuje i ubrzava proces obnove. Automatizirana obnova poslužitelja je puno brža za razliku od ručne instalacije aplikacije i ponovnog postavljanja svih zahtjeva sustava.

Proces obnove poslužitelja počinje sa instalacijom operacijskog sustava. Točna verzija potrebnog operacijskog sustava i jezgre se može očitati iz `server-info` datoteke. Ostatak obnove poslužitelja i aplikacije se može izvršiti preko skripte `complete-restore.sh` (Ispis 8). Prije pokretanja skripte je potrebno sigurnosnu kopiju kopirati na novi poslužitelj na putanju `/home/<username>/Backup/` (kreirati direktorij ako ne postoji).

```
#! /bin/bash
BACKUP_DIR=home/$USER/Backup

# 1) Obnova APT repozitorija
sudo wget https://dev.mysql.com/get/mysql-apt-config_0.8.22-1_all.deb
sudo dpkg -i mysql-apt-config_0.8.22-1_all.deb
sudo apt update
# 2) Obnova debian paketa
sudo apt-get install $(cat /$BACKUP_DIR/packages_list_backup)
# 3) Obnova konfiguracijskih datoteka
sudo rsync -av /$BACKUP_DIR/configs/apache2 /etc/
sudo rsync -av /$BACKUP_DIR/configs/mysql /etc/
sudo rsync -av /$BACKUP_DIR/configs/php /etc/
sudo rsync -av /$BACKUP_DIR/configs/phpmyadmin /etc/
sudo a2enconf phpmyadmin
# 4) Obnova datoteka Prestashop aplikacije
sudo rm /var/www/html/*
sudo rsync -av /$BACKUP_DIR/prestashop-app-files/ /var/www/html/
# 5) Obnova podataka u bazu podataka
sudo mysql -u root -p < /$BACKUP_DIR/full-backup-*.sql
# 6) Ponovo pokretanje web servisa
sudo systemctl restart apache2
```

Ispis 8: Skripta za obnovu poslužitelja

Skripta na poslužitelju radi sljedeće:

1. Dodavanje APT repozitorija za pakete koji to zahtijevaju
2. Obnova paketa operacijskog sustava preko APT alata
3. Obnova konfiguracijskih datoteka u potrebne direktorije
4. Obnova datoteka Prestashop aplikacije
5. Obnova baze podataka preko uvoza SQL *dump* datoteke
6. Ponovno pokretanje `apache2` servisa da bi se primijenile promjene datoteka i postavki

Pokretanjem skripte terminal postaje interaktivan te korisnik mora pratiti upute instalacijskih pomoćnika za dovršetak instalacije alata. Nakon dovršetka skripte obnova poslužitelja je gotova i aplikacija je spremna za rad.

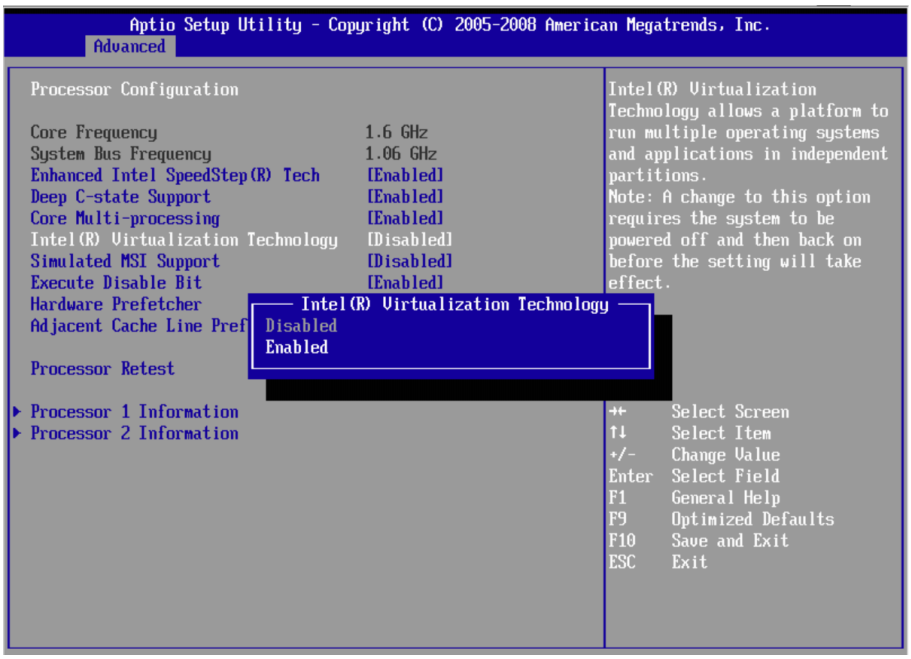
Potencijalna mana ovog pristupa je promjena dostupnih alata u inicijalnim Debian APT repozitorijima. Primjerice, može se dogoditi da nakon dvije godine kad se pokrene skripta `complete-restore.sh` (Ispis 8) u drugom koraku ne može pronaći paket `php7.4` jer je u međuvremenu puštena novija stabilna verzija `php8.0` i zamijenila se sa verzijom `7.4` u Debian repozitoriju. Kako bi skripta ispravno radila potrebno je ručno dodati PHP repozitorij u APT repozitorije slično kako se to učinilo za MySQL paket u prvom koraku i tad bi se problem otklonio.

4. Virtualni stroj

U poglavlju je opisano postavljanje okruženja poslužitelja za rad Prestashop aplikacije unutar virtualnog stroja. Na poslužitelju je instaliran Debian 11 operacijski sustav. Opisana je instalacija QEMU virtualizatora i KVM hipervizora unutar Debian operacijskog sustava. Naposljetku je objašnjen proces izrade sigurnosne kopije u ovakvom okruženju te oporavak samog poslužitelja.

4.1. Postavljanje virtualnog stroja

Prije same instalacije softvera virtualizatora u postavkama BIOS-a (*Basic Input/Output System*) poslužitelja je potrebno omogućiti tehnologiju virtualizacije. Svaki poslužitelj ima specifične korake za promijeniti tu postavku stoga nema univerzalnih uputa kako to učiniti. Na slici (Slika 18) se može vidjeti primjer kako izgleda zaslon sa postavkom virtualizacije.



Slika 18: Postavka za omogućiti virtualizaciju [15]

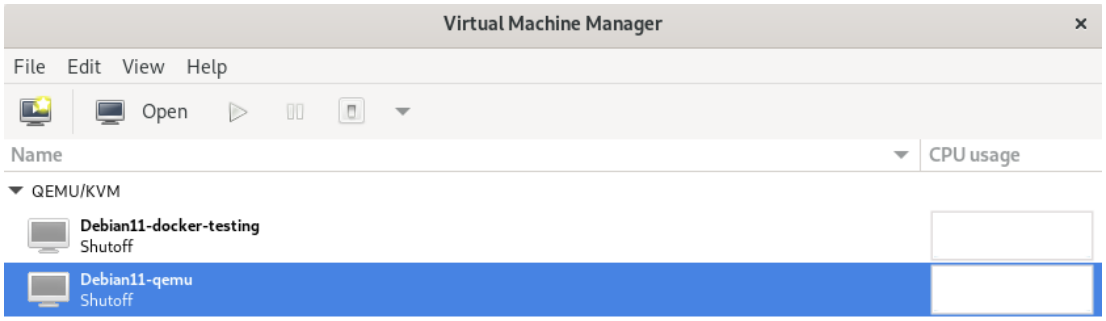
Instalacija virtualizacijskog softvera se može obaviti preko APT alata kao i prijašnji alati. Instalacija potrebnih modula i paketa se može odraditi preko naredbe `sudo apt install qemu-system libvirt-clients libvirt-daemon-system virt-manager`. `Qemu-system` paket sadrži binarne datoteke za virtualizaciju sustava.

Libvirt je alat napisan u C jeziku koji služi za interakciju sa virtualizacijskim mogućnostima operacijskog sustava [16]. Cilj biblioteke je pružiti stabilan API (*Application Programming Interface*) za različite mehanizme virtualizacije. Paket `libvirt-clients` sadrži `libvirt` ljusku (engl. *Shell*) `virsh` koja služi za upravljanje virtualnim strojevima putem terminala. Paket `libvirt-daemon-system` sadrži konfiguracijske datoteke koje su potrebne za pokretanje `libvirt` procesa kao sistemskog procesa.

`Virt-manager` (*Virtual Machine Manager*) paket sadrži aplikaciju sa grafičkim korisničkim sučeljem koja olakšava upravljanje virtualnim strojevima putem Libvirt-a. Također omogućuje prikaz aktivnih virtualnih strojeva, njihove izvedbe i statistiku korištenja resursa.

Nakon instalacije je potrebno dodati korisnika u `libvirt` grupu pomoću naredbe `sudo adduser $USER libvirt` da bi korisnik mogao upravljati virtualnim strojevima.

Kreiranje virtualnog stroja se može odraditi preko virt-manager aplikacije. Otvaranjem aplikacije se dobije početni zaslon (Slika 19):



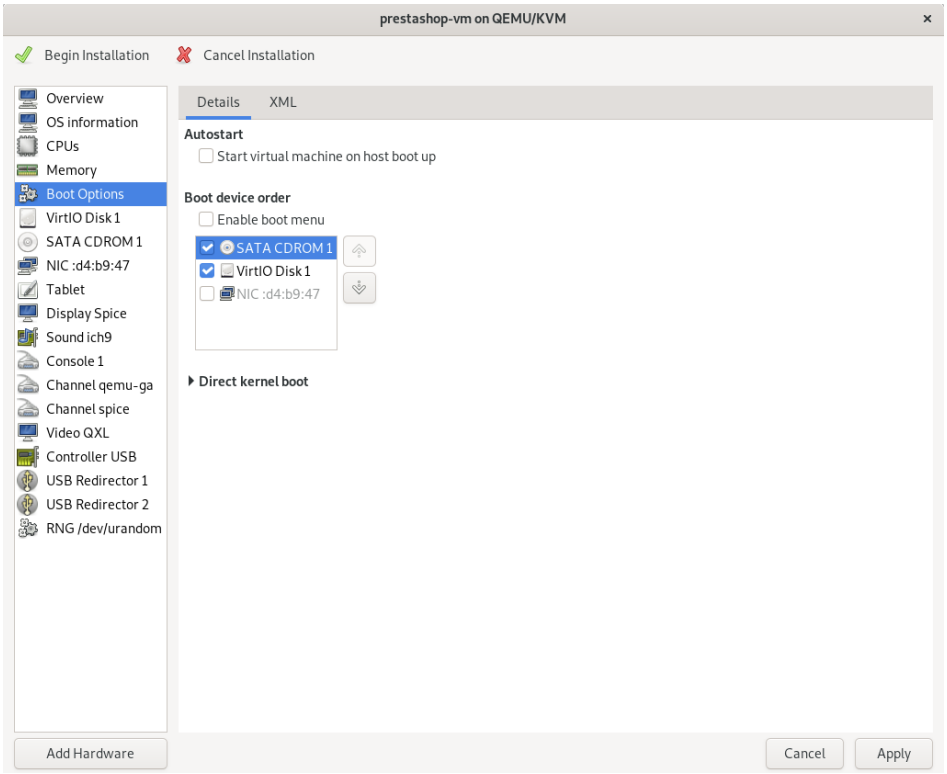
Slika 19: *Virtual Machine Manager* grafičko korisničko sučelje

Za izradu virtualnog stroja potrebno je pratiti korake „čarobnjaka“ za instalaciju (engl. *Installation wizard*) nakon pritiska na gumb „*New Virtual Machine*“. Čarobnjak omogućuje stvaranje novih virtualnih strojeva te konfiguraciju i raspodjele resursa stroja i virtualnog hardvera.

Za instalaciju Debian 11 operacijskog sustava je potrebna ISO (*Image Standard Optical*) *image* datoteka koju je moguće preuzeti sa službene Debian web stranice https://www.debian.org/download. Nakon preuzimanja datoteke na poslužitelj potrebno je putanju datoteke dati čarobnjaku da bi instalirao operacijski sustav. Osim toga potrebno je postaviti određene parametre za rad virtualnog stroja:

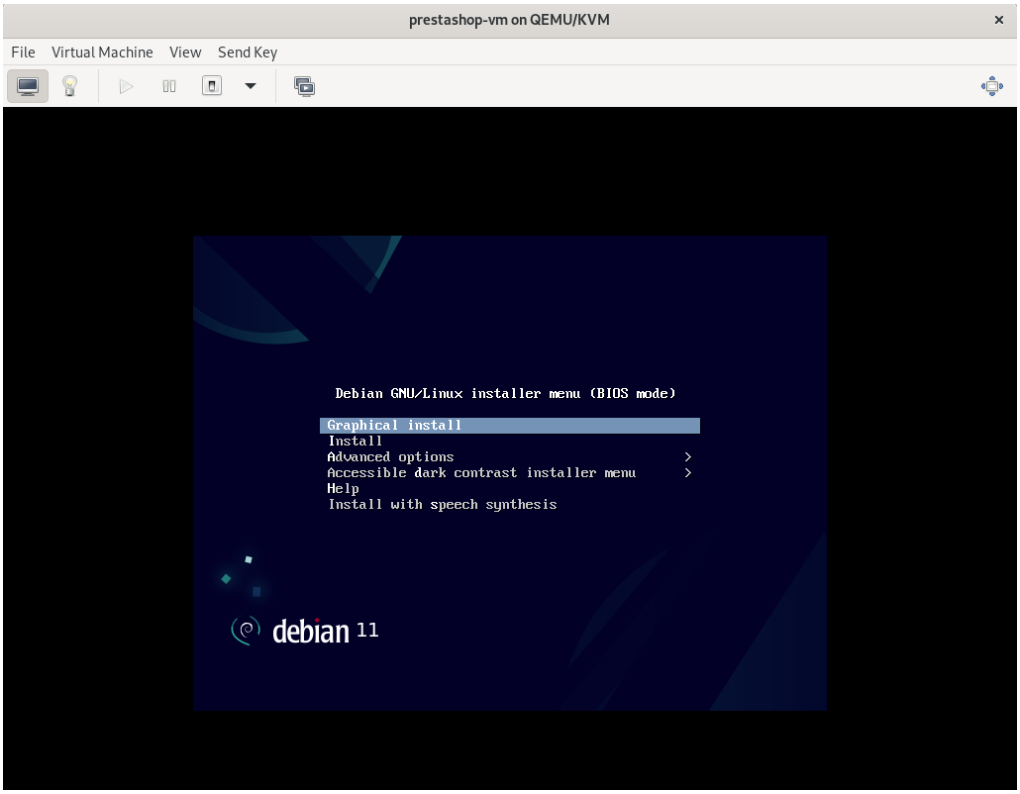
- Memorija (RAM) – preporučeno minimalno 2048 MB
- Broj virtualnih procesora (*CPUs*) – preporučeno minimalno 2
- Veličina pohrane (*disk image*) – preporučeno minimalno 40 GB
- Ime virtualnog stroja – primjerice `prestashop-vm`

Na zadnjem zaslonu čarobnjaka potrebno je kliknuti na potvrdni okvir (engl. *Checkbox*) „*Customize configuration before install*“. Nakon što je čarobnjak završio otvoriti će se postavke virtualnog stroja te potrebno je otvoriti postavke „*Boot options*“ (Slika 20). Treba se postaviti točan redoslijed pokretanja uređaja na stroju tako da se pokrene Debian 11 *image* datoteka za instalaciju operacijskog sustava.



Slika 20: Postavke za redoslijed pokretanja uređaja

Nakon toga se može kliknuti na gumb „*Begin Installation*“ te će se nakon instalacije otvoriti prozor sa pokrenutom virtualnim strojem `prestashop-vm` (Slika 21).



Slika 21: Virtualni stroj `prestashop-vm`

Sada je virtualni stroj podignut i spreman je za instalaciju Debian operacijskog sustava. Koraci instalacije operacijskog sustava će biti isti kao i koraci za fizički poslužitelj. Prethodno, u poglavljima 3.1, 3.2, 3.3, 3.4 su opisani koraci postavljanja poslužitelja i konfiguracija, izrada baze podataka te naposljetku instalacija same Prestashop Aplikacije. Činjenicom da su koraci identični za virtualni poslužitelj kao u nabrojanim poglavljima, ti koraci se neće ponovno opisivati već se mogu pratiti iz navedenih poglavlja.

4.2. Izrada sigurnosne kopije

Prestashop aplikacija sada radi unutar virtualnog stroja stoga je sigurnosnu kopiju moguće izraditi na dva načina:

1. Na razini virtualnog stroja
2. Na razini operacijskog sustava unutar virtualnog stroja (gostujući operacijski sustav)

4.2.1 Sigurnosna kopija na razini virtualnog stroja

Sav sadržaj i podaci virtualnog stroja se spremaju na *host* poslužitelju u datoteku tipa `qcow2` (*QEMU copy-on-write 2*). Takav način rada stroja omogućava izradu sigurnosne kopije cijelog virtualnog stroja preko samo jedne datoteke. U toj jednoj datoteci su svi alati, moduli, konfiguracije i datoteke gostujućeg operacijskog sustava što olakšava proces izrade sigurnosne kopije. Oporavak takve sigurnosne kopije je jednostavan jer tijekom procesa obnove je potrebno samo uvesti (engl. *Import*) virtualni stroj kao cjelinu u jednom koraku. Nakon obnove potrebno je samo pokrenuti virtualni stroj i Prestashop aplikacija će biti spremna za rad jer je sve već podešeno.

Takvu izradu sigurnosne kopije je moguće izraditi dok je virtualni stroj ugašen ili podignut. Razlika je u stanju virtualnog stroja dok se vrši izrada sigurnosne kopije. Hladna (engl. *Cold*) izrada sigurnosne kopije je izrada sigurnosne kopije dok je virtualni stroj spušten, tj. ugašen. Takva izrada sigurnosne kopije je sigurnija jer se operacijski sustav i baza podataka ugase na normalan predviđen način. Drugim riječima, svi podaci koji su trebali biti zapisani su zapisani na disk po predviđenom načinu bez prekida.

Suprotno tome, vruća (engl. *Hot*) izrada sigurnosne kopije je izrada sigurnosne kopije dok je virtualni stroj podignut, tj. dok se izvršava. Takva izrada sigurnosne kopije je moguća, ali je manje sigurna za očuvanost podataka. Podaci unutar takve sigurnosne kopije mogu biti pokvareni (engl. *Corrupted*) ili nedosljedni. No ako se radi na ispravan način može se osigurati da je izrada takve kopije minimalnog rizika.

U ovom poglavlju će se opisati vruća izrada sigurnosne kopije jer je pogodnije da je Prestashop web aplikacija uvijek dostupna kupcu.

Izrada sigurnosne kopije dok je virtualni stroj podignut je moguća pomoću `virsh` klijenta preko sljedećih koraka:

- Spremanje konfiguracije virtualnog stroja
- Kreiranje vanjske snimke (engl. *External snapshot*) virtualnog stroja (kreiranje *overlay* datoteke)
- Izrada sigurnosne kopije bazične *image* datoteke (`qcow2`) virtualnog stroja

Nakon izrade sigurnosne kopije je moguće sjediniti (engl. *Merge*) *overlay* datoteku sa bazičnom datotekom. Navedene korake je moguće izvršiti preko skripte `live-vm-backup.sh` (Ispis 9).

```
#! /bin/bash
CURRENT_DATE=$(date +"%d_%m_%Y")
sudo mkdir -p /vm-backups/$CURRENT_DATE

# 1) Kreiraj kopiju konfiguracijske datoteke vm-a
sudo virsh dumpxml prestashop-vm > /vm-backups/$CURRENT_DATE/prestashop-
vm.xml
# 2) Napravi vanjsku snimku, tj. kreiraj overlay datoteku imena
prestashop-vm-overlay1.qcow2
sudo virsh snapshot-create-as --domain prestashop-vm overlay1 \
--diskspec hda,file=/var/lib/libvirt/images/prestashop-vm-overlay1.qcow2
\
--disk-only \
--atomic
# 3) Izradi sigurnosnu kopiju bazične image datoteke
sudo tar -Sczvf /vm-backups/$CURRENT_DATE/prestashop-vm-image-
backup.tar.gz -C /var/lib/libvirt/images prestashop-vm.qcow2
# 4) Sjedini sadržaj overlay1 datoteke i bazične image datoteke
sudo virsh blockcommit prestashop-vm hda --active --pivot
# 5) izbriši konfiguracijsku datoteku vanjske snimke imena "overlay1"
sudo virsh snapshot-delete prestashop-vm overlay1 --metadata
sudo rm /var/lib/libvirt/images/*overlay*
# 6) Kopiraj sigurnosnu kopiju na sigurnu lokaciju
sudo rsync -av /vm-backups/$CURRENT_DATE /media/$USER/external_drive/
```

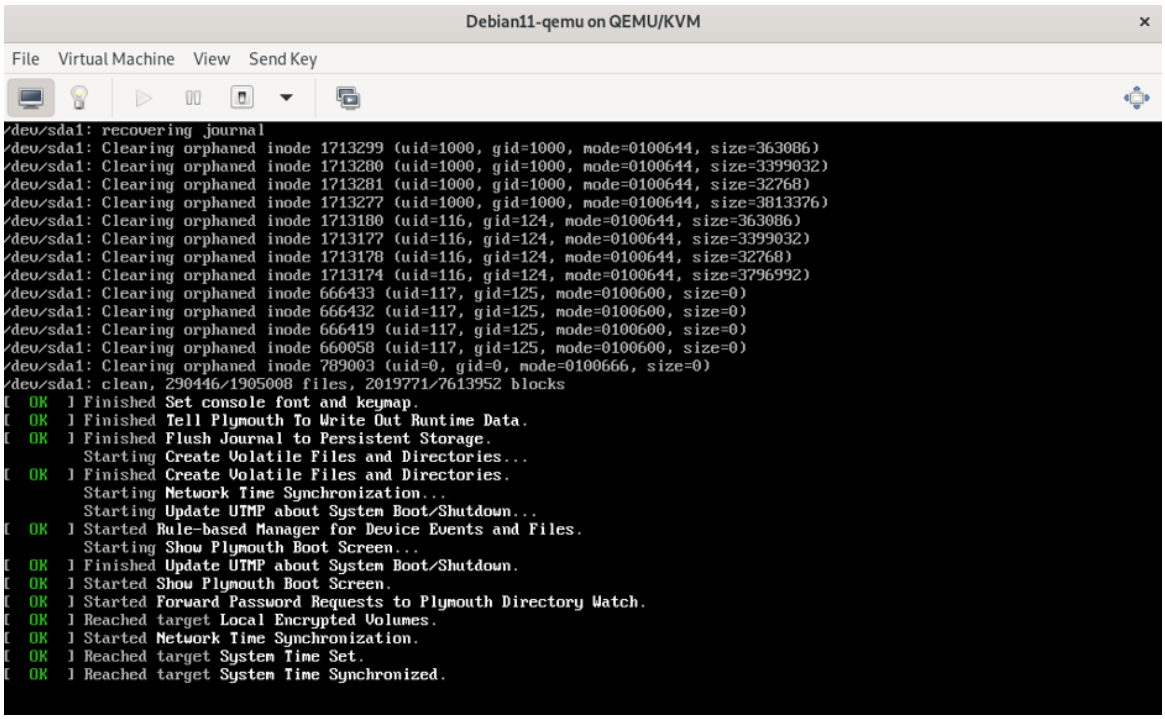
Ispis 9: Skripta za izradu sigurnosne kopije virtualnog stroja

U konfiguracijskoj datoteci je spremljena sama konfiguracija virtualnog stroja. U njoj su zapisani meta podaci koji opisuju virtualni stroj i potrebna je prilikom rekreiranja i podizanja stroja. Činjenicom da je potrebna prilikom rekreiranja, u prvom koraku želi se izraditi kopija sa naredbom `virsh dumpxml prestashop-vm > prestashop-vm.xml`. Pomoću `virsh dumpxml` naredbe se zapisuje konfiguracija `prestashop-vm` virtualnog stroja u XML (*Extensible Markup Language*) datoteku `prestashop-vm.xml`.

Zbog dosljednosti podataka prilikom izrade sigurnosne kopije podignutog virtualnog stroja koristi se tehnologija izrađivanja snimaka. Snimke uzimaju disk, memoriju i stanje uređaja virtualnog stroja u određenom trenutku i spremaju ih za upotrebu. Podignuti virtualni stroj aktivno zapisuje promjene na disk stoga je za izradu sigurnosne kopije potrebno koristiti snimke stroja u određenom trenutku. Kad bi se radila sigurnosna kopija podataka na aktivnom stroju za koju nije napravljena snimka dogodilo bi se da je posljedična sigurnosna kopija nedosljedna. Kopija bi bila nedosljedna jer bi se takva kopija jednim dijelom imala stare podatke, a drugim dijelom nove podatke. Takva sigurnosna kopija je nepouzdana i ne bi se trebala koristiti.

Da se riješi problem nedosljednosti podataka obaviti će se vanjska snimka virtualnog stroja. To znači da se u postojeću bazičnu *image* datoteku koja sadrži podatke virtualnog stroja neće moći zapisivati nove promjene (*read-only*) već će se kreirati nova datoteka (*overlay*). *Overlay* datoteka će biti povezana na bazičnu datoteku i preko nje će se moći čitati sadržaj iz bazične datoteke. Također, u *overlay* datoteku će se zapisivati nove promjene koje se događaju na virtualnom stroju i stoga se može izraditi sigurnosna kopija bazične *image* datoteke koja ne mijenja stanje tijekom kopiranja. U drugom koraku se pomoću naredbe `virsh snapshot-create-as` odrađuje opisani proces izrade vanjske snimke. Datoteka `prestashop-vm-overlay1.qcow2` će biti novoizrađena *overlay* datoteka u koju će se zapisivati nove promjene virtualnog stroja.

Kako bi se izbjeglo „zamrzavanje“ (engl. *Stun*) virtualnog stroja koristi se parametar `–disk-only` prilikom izrade snimke. Pomoću parametra se nalaže da će datoteka kreirana snimkom uključivati samo sadržaj diska, a ne uobičajenu snimku cijelog sustava sa stanjem virtualnog stroja. Bez tog parametra u datoteku bi se zapisivali i podaci iz RAM-a te bi tijekom procesa zapisivanja virtualni stroj bio zaustavljen, tj. zamrznut, a takvo ponašanje poslužitelja nije poželjno. Snimke diska hvataju se brže (gotovo trenutno) nego snimke cijelog sustava, ali oporavak takve snimke će biti kao ponovno uključivanje sustava koji je naglo ostao bez napajanja i nije stigao uraditi proces gašenja. Stoga će se prilikom obnove sustava najčešće morati izvršiti oporavak dnevnika (engl. *Journal recovery*) operacijskog sustava (Slika 22) i MySQL oporavak od pada. Postoji mogućnost da se sustav neće moći oporaviti iz takvog stanja i stoga je takva sigurnosna kopija nesigurna. Takvo stanje sigurnosne kopije se naziva sigurnosna kopija dosljedna rušenju (engl. *Crash-consistent backup*).



Slika 22: Oporavak dnevnika operacijskog sustava

U naredbi je naveden i `-atomic` parametar koji jamči da se neće izvršiti nikakva promjena u slučaju neuspjeha prilikom kreiranja snimke.

U trećem koraku se kreira arhiva u koju se sprema bazična *image* datoteka virtualnog stroja `prestashop-vm`. Radi dodatne optimizacije koristi se parametar `-S` i `-z`. Pomoću `-z` (`--gzip`) parametra vrši se sabijanje (engl. *Compress*) datoteka u arhivu pomoću `gzip` alata. Osim toga koristi se i parametar `-S` (`--sparse`) za učinkovito arhiviranje *sparse* datoteka. *Sparse* datoteka je datoteka u datotečnom sustavu koja u sebi ima „rupe“. Rupa u datoteci je dio sadržaja datoteke koji nikad nije zapisan u datotečni sustav. To znači ako primjerice datoteka ima duljinu 40GB, a popunjeno je samo 10GB sadržaja onda će se u datotečnom sustavu dinamički alocirati samo 10GB i toliko će mjesta zauzimati na disku. Upravo takva datoteka je *image* datoteka virtualnog stroja `prestashop-vm` u koju se sprema sadržaj stroja. Koristeći `-sparse` parametar se postiže da arhiva zauzima samo onoliko mjesta na disku koliko je dinamički alocirano za *image* datoteku. Sa time se postiže da je na disku poslužitelja puno više mjesta slobodno prilikom izrade sigurnosne kopije.

Kada se arhiva uspješno izradila, nema više potrebe za dodatnom *overlay* datotekom. Stoga se može sadržaj bazične *image* datoteke sjediniti sa sadržajem *overlay* datoteke. Taj proces se u četvrtkom koraku vrši pomoću naredbe `virsh blockcommit`.

U petom koraku se brišu konfiguracijska datoteka `overlay1.xml` i sama `overlay` datoteka koje su se kreirale prilikom izrade vanjske snimke.

Naposljetku, u šestom koraku se sigurnosna kopija virtualnog stroja kopira na sigurnu lokaciju.

4.2.2 Sigurnosna kopija na razini gostujućeg operacijskog sustava

U poglavlju 4.2.1 je opisan potencijalan rizik prilikom obnove takvom metodom. Za razliku od tog pristupa, moguće je napraviti sigurnosnu kopiju na razini gostujućeg operacijskog sustava koja nema spomenute rizike.

Virtualni stroj `prestashop-vm` se može gledati kao običan fizički poslužitelj sa operacijskim sustavom Debian. Stoga je moguće izraditi sigurnosnu kopiju na isti način kao u poglavlju 3.5. Sigurnosna kopija se može izravno kopirati sa gostujućeg operacijskog sustava na neku sigurnu lokaciju.

4.3. Automatizacija pomoću cron alata

Automatizacija izrade sigurnosne kopije je moguća za obje opisane metode sigurnosne kopije. Dapače, zbog lakše, sigurnije i brže obnove virtualnog stroja se može preporučiti automatizacija za oba načina izrade sigurnosne kopije. Jedina mana takvog pristupa je što će se utrošiti dodatna pohrana za spremanje sigurnosne kopije no to ne predstavlja problem jer se ne radi o velikim količinama podataka.

Kako je sigurnosna kopija na razini virtualnog stroja puno veća od druge sigurnosne kopije jer se kopira cijeli virtualni poslužitelj, ne treba je izrađivati svaki dan. To je u redu jer se sami operacijski sustav, alati i moduli ne mijenjaju toliko često stoga se ne treba svakodnevno izrađivati kopija za te dijelove sustava. Može se izraditi cron posao za `live-vm-backup.sh` skriptu (Ispis 9) koja će se izvršavati jednom tjedno u nedjelju u 04:00 (Ispis 10).

```
0 4 * * SUN /backup-scripts/live-vm-backup.sh
```

Ispis 10: Cron posao za automatizaciju izrade sigurnosne kopije virtualnog stroja

Slično kao u poglavlju 3.5.1 može se postaviti cron posao za `complete-backup.sh` skriptu (Ispis 5) koja će se izvršavati jednom dnevno u 03:00 (Ispis 11).

```
0 3 * * * /backup-scripts/complete-backup.sh
```

Ispis 11: Cron posao za automatizaciju izrade sigurnosne kopije na razini gostujućeg operacijskog sustava

Na ovaj način će na sigurnoj lokaciji biti dostupne obje sigurnosne kopije. Prednost ovakvog pristupa se može vidjeti u poglavlju 4.5.

4.4. RAID1 zrcaljenje

U poglavlju 2.8 je opisano korištenje RAID1 konfiguracije diskova kako bi se postigla dodatna sigurnost i redundancija podataka. RAID1 konfiguracija će biti moguća i preporučena za poslužitelj sa virtualizatorom. Činjenicom da se sami virtualni stroj i njegovi podaci i sadržaj spremaju na *host* poslužitelj ima smisla napraviti zrcaljenje diska na kojem je operativni sustav *host* poslužitelja. Postupak izrade RAID1 konfiguracije će biti identičan kao u poglavlju 3.5.2 jer se konfiguracija radi za *host* poslužitelj.

4.5. Obnova virtualnog stroja

Obnova virtualnog stroja se može odraditi na tri načina:

1. Obnova sigurnosne kopije cjelokupnog virtualnog stroja
2. Obnova sigurnosne kopije na razini gostujućeg operacijskog sustava
3. Kombinacija prva dva načina

Kao što je prethodno opisano, prvi način je vrlo jednostavan za izvršiti jer je cijeli sadržaj stroja spremljen u jednu datoteku. Na novi poslužitelj je potrebno instalirati Debian 11 operacijski sustav i softver za virtualizaciju kako je opisano u poglavlju 4.1. Nakon toga se sa vanjske pohrane pomoću `rsync` naredbe na poslužitelj kopira sigurnosna kopija u direktorij `/var/lib/libvirt/images`. Jedino je preostalo pokrenuti skriptu `vm-restore.sh` (Ispis 12) koja će *image* datoteku virtualnog stroja izvući iz arhive u potreban direktorij i definirati će konfiguracijsku datoteku za taj stroj preko naredbe `virsh define`. Samostalno, ovaj pristup je brz, ali se može dogoditi da se izgubilo podataka za više dana jer se sigurnosna kopija izvršava samo jednom tjedno.

```
#! /bin/bash
```

```
sudo cd /var/lib/libvirt/images/
sudo tar -xzvf prestashop-vm-image-backup.tar.gz
sudo virsh define prestashop-vm.xml
```

Ispis 12: Skripta za obnovu virtualnog stroja

Kao i za prvi način, za drugi način je potrebno instalirati Debian 11 operacijski sustav i softver za virtualizaciju. Nadalje, potrebno je ručno kreirati novi virtualni stroj te instalirati još jednom Debian 11 operacijski sustav kako je objašnjeno u poglavlju 4.1. Nakon što je instaliran operacijski sustav na virtualnom stroju potrebno je pratiti korake za obnovu poslužitelja u poglavlju 3.6. Ovaj način zahtijeva više ručnog rada od prvog te je potrebno više vremena za taj rad.

Preko trećeg načina moguće je kombinirati obnovu sigurnosne kopije prvog i drugog pristupa. U prvom pristupu se spomenulo da se može dogoditi da se izgube podaci za više dana jer se sigurnosna kopija izrađuje samo jednom tjedno. Jedan način za riješiti taj problem je da se poveća interval izvršavanja sigurnosne kopije no onda će se vanjska pohrana puno brže popuniti. Elegantno rješenje tog problema je koristiti kombinaciju sigurnosnih kopija na razini virtualnog stroja i gostujućeg operacijskog sustava.

Primjerice, zamislimo da se poslužitelj pokvario u srijedu ujutro u 09:00. Zadnja sigurnosna kopija virtualnog stroja je izrađena u nedjelju u 04:00 te sigurnosna kopija gostujućeg operacijskog sustava u srijedu u 03:00. Ako koristimo prvi način obnove, onda će se izgubiti svi podaci koji su zapisani između nedjelje u 04:00 i srijede u 09:00, tj. izgubiti će se podaci zapisani za cijeli ponedjeljak i utorak i dio srijede. Međutim, nakon obnove virtualnog stroja prvim pristupom možemo se poslužiti sigurnosnom kopijom koja se spremila sa gostujućeg operacijskog sustava. Pretpostavimo da se nije mijenjalo okruženje poslužitelja, alati i moduli ni datoteke Prestashop aplikacije već samo podaci unutar baze podataka. Tada se može u obnovljeni virtualni stroj `mysqldump` alatom obnoviti podaci za ponedjeljak i utorak te srijedu do 03:00 preko *dump* datoteke koja se nalazi u sigurnosnoj kopiji. Ovim pristupom se gube samo podaci koja su zapisani u srijedu između 03:00 i 09:00, tj. gube se samo podaci za šest sati.

Nadalje, ovim kombiniranim pristupom se ne mora brinuti o dostupnim paketima u APT repozitorijima jer su se svi paketi i alati obnovili sa cjelokupnim virtualnim strojem.

Zaključno, ovakva kombinirana sigurnosna kopija se pokazala efikasna i djelotvorna te dodatno sprječava gubitak podataka i osigurava fleksibilnu obnovu poslužitelja.

5. Docker

U ovom poglavlju je prikazan rad Prestashop aplikacije na Docker platformi. Opisano je izvođenje aplikacije unutar Docker spremnika i proces rada koji omogućuje rad aplikacije u takvom okruženju. Naposljetku je objašnjen način izrade sigurnosne kopije važnih dijelova aplikacije te oporavak istih. Na *host* poslužitelj je instaliran Debian 11 operacijski sustav te se na njemu nalazi Docker.

Nakon pokretanja spremnika, proces instalacije Prestashop aplikacije će biti isti kao i dio sa pomoćnikom za instalaciju u poglavlju 3.4. Postojati će samo jedna razlika, umjesto vrijednosti `127.0.0.1` za parametar *Database server address* treba staviti vrijednost `mysql`.

5.1. Postavljanje Docker plaftorme

Kako bi se Prestashop aplikacija izvodila na Docker platformi potrebno je kreirati Docker *image* datoteku. Činjenicom da Prestashop aplikacija sadrži više tehnologija i veći broj komponenti koje međusobno komuniciraju, preporučeno je napraviti više *image* datoteka koje će kreirati više spremnika koji će moći međusobno komunicirati.

Da bi se olakšalo kreiranje tih datoteka i spremnika, koristiti će se Docker Compose alat. Docker Compose omogućava kreiranje i pokretanje svih servisa iz YAML (*YAML Ain't Markup Language*) konfiguracijske datoteke preko samo jedne naredbe. Tu datoteku korisnik svojeručno kreira i definira te je pruža Docker Compose alatu za učitavanje. Za kreiranje i podizanje Prestashop aplikacije koristi se YAML datoteka imena `docker-compose.yml` (Ispis 13).

```
version: '3.9'

services:
    mysql:
        image: mysql:5.7
        container_name: prestashop-db
        environment:
            MYSQL_DATABASE: prestashopdb
            MYSQL_ROOT_PASSWORD: lozinka123
        ports:
            - 3307:3306
        volumes:
            - prestashop-data:/var/lib/mysql
            - mysql-config:/etc/mysql
        networks:
```

```
            - prestashop-net

    prestashop:
        image: prestashop/prestashop:1.7
        container_name: prestashop-app
        environment:
            DB_SERVER: mysql
        ports:
            - 8080:80
        volumes:
          - prestashop-files:/var/www/html
          - apache2-config:/etc/apache2
          - php-config:/etc/php
        networks:
            - prestashop-net
        depends_on:
            - mysql

    phpmyadmin:
        image: phpmyadmin/5.2
        container_name: phpmyadmin
        environment:
            PMA_HOST: mysql
        ports:
            - 1235:80
        volumes:
            - pma-config:/etc/phpmyadmin
        networks:
            - prestashop-net
        depends_on:
            - mysql

networks:
    prestashop-net:
volumes:
    prestashop-files:
    mysql-config:
    apache2-config:
    php-config:
    pma-config:
```

Ispis 13: `docker-compose.yml` konfiguracijska datoteka

Iz ispisa (Ispis 13) se može vidjeti da su definirana tri servisa:

1. `Mysql`
2. `Prestashop`
3. `Phpmyadmin`

To znači da će se kreirati ukupno tri Docker spremnika te će svaki spremnik imati jedinstvenu svrhu. `Mysql` spremnik će služiti kao baza podataka koja komunicira sa Prestashop aplikacijom i pruža joj podatke iz MySQL baze podataka. `Prestashop`

spremnik će sadržavati samu Prestashop aplikaciju, tj. njene datoteke i također Apache HTTP poslužitelj te tumač jezika PHP. `Phpmyadmin` spremnik sadrži PhpMyAdmin alat koji se koristi za administraciju MySQL baze podataka preko grafičkog korisničkog sučelja.

5.2. Docker *image*

Konfiguracija za `mysql` servis specificira da će Docker Compose kreirati i pokrenuti Docker spremnik imena `prestashop-db` na temelju Docker *image* datoteke imena `mysql:5.7`. Drugim riječima, prilikom izvršavanja Docker naredbe za stvaranje i pokretanje servisa prema konfiguraciji (`docker compose up`), Docker će za `mysql` servis pregledati upute `mysql:5.7` službene *image* datoteke te će po njenim instrukcijama kreirati i pokrenuti `mysql` spremnik. Ako Docker na lokalnom poslužitelju ne može pronaći *image* datoteku onda će je potražiti u Docker Hub repozitoriju.

Kada se nalaže koji Docker *image* se želi koristiti, potrebno je u konfiguraciji specificirati ime prema šabloni `ime_repozitorija:tag`. Sa time Docker zna da mora unutar `mysql` repozitorija pronaći *image* oznake (engl. Tag) `5.7`. Službena `mysql` Docker dokumentacija kaže da će za tu *image* datoteku Docker dobiti upute da kreira spremnik koji će sadržavati MySQL sustav verzije 5.7.

Slično kao i za `mysql` servis, u `prestashop` servisu je također definiran službeni Docker *image* `prestashop/prestashop:1.7` kojeg je kreirao Prestashop tim. Koristi se *image* oznake `prestashop:1.7` koji sadržava upute za kreiranje spremnika koji sadrži Prestashop aplikaciju verzije 1.7. Naposljetku, isti princip je primijenjen i za `phpmyadmin` servis koji koristi PhpMyAdmin alat verzije 5.2.

5.3. Docker volumeni

Po standardnome ponašanju Docker ne očuva podatke nakon gašenja i uništenja Docker spremnika. Činjenicom da je Prestashop web aplikacija koja obavlja funkciju prodavaonice, postoje razni podaci koji se žele trajno sačuvati kako bi se postigao optimalan i uspješan rad aplikacije. U poglavlju 3.5 je već prethodno opisano za koje podatke se pravi sigurnosna kopija, tj. koji podaci se trebaju očuvati.

Kako bi se očuvali podaci potrebni za rad aplikacije koriste se Docker volumeni. Zbog korištenja Docker volumena na *host* poslužitelju će se spremiti datoteke i podaci Docker spremnika za koje su napravljeni volumeni. Docker volumeni, tj. specifično Docker

imenovani volumeni (engl. *Named volumes*) se definiraju po šabloni `ime_volumena:odredište` unutar `docker-compose.yml` datoteke. `Ime_volumena` označava naziv volumena koji će se dodijeliti tom volumenu, a `odredište` definira putanju (engl. *Path*), tj. gdje će se datoteka ili direktorij montirati (engl. *Mount*) unutar spremnika. Drugim riječima, definiranjem odredišta određujemo koje će se datoteke i direktoriji očuvati na *host* poslužitelju.

Prethodno je već definirano za koje se podatke izrađuje sigurnosna kopija stoga je upravo za te direktorije i podatke potrebno napraviti Docker volumene. Volumeni se nabrajaju pod `volumes` dijelom. Također je potrebno na dnu datoteke zapisati sve imenovane volumene pod parametrom `volumes` čije je uvlačenje (engl. *Indentation*) na razini `services` parametra. Docker volumeni koji se koriste za potrebe aplikacije su:

- `Prestashop-data` (`/var/lib/mysql`)
- `Mysql-config` (`/etc/mysql`)
- `Prestashop-files` (`/var/www/html`)
- `Apache2-config` (`/etc/apache2`)
- `Php-config` (`/etc/php`)
- `Pma-config` (`/etc/phpmyadmin`)

Zaključno, zbog definiranja navedenih Docker volumena prilikom gašenja spremnika ti podaci i direktoriji se neće uništiti jer će se trajno spremati na *host* poslužitelj. Prilikom ponovnog pokretanja spremnika Docker će biti svjestan postojećih volumena i njihov sadržaj će montirati u spremnik i stoga će ti podaci biti dostupni i neće biti izgubljeni.

5.4. Preostala konfiguracija

Pomoću `container_name` se određuje ime spremnika koji će biti kreiran. Ako se `container_name` ne zada, Docker će automatski generirati ime spremnika no zbog jednostavnijeg rukovođenja spremnikom preporuča se korisniku da mu zada ime.

Kako bi se `mysql` spremnik ispravno kreirao, također je potrebno zadati određene varijable okoline (engl. *Environment variables*). Varijable okoline služe za postavljanje varijabli unutar spremnika ili za gaženje (engl. *Overwrite*) varijabli koje su definirane u

Docker *image* datoteci. Varijable okoline pružaju jednostavno i efikasno mijenjanje konfiguracije za spremnike.

Službena Docker `mysql`, `prestashop` i `phpmyadmin` dokumentacija nalaže da postoje varijable okoline koje su obavezne i neobavezne za kreiranje Docker spremnika. Za `mysql` spremnik jedina obavezna varijabla okoline je `MYSQL_ROOT_PASSWORD` i preko nje se specificira lozinka koja će biti postavljena za MySQL *root* korisnički račun. Postoji više neobaveznih varijabli okoline za `mysql` *image*, no za potrebe projekta definirana je samo `MYSQL_DATABASE` varijabla koja omogućava da se odredi ime baze podataka koja će se stvoriti prilikom pokretanja spremnika. Ukoliko ta baza podataka već postoji, onda će preskočiti taj korak.

Obaveznih varijabli okolina nema za `prestashop` spremnik, no u konfiguraciji se definira `DB_SERVER` varijabla koja određuje ime vanjske (engl. *External*) MySQL baze podataka koja će se koristiti za spremanje podataka aplikacije. Postoje mnogo ostalih varijabli okoline, no sve su neobavezne i nisu potrebne za ovaj rad. Slično kao i `prestashop` spremnik, `phpmyadmin` spremnik ne zahtijeva obavezne varijable okoline već se koristi samo `PMA_HOST` varijabla da bi se odredilo na koji vanjski MySQL poslužitelj se spremnik mora spojiti.

Nadalje, potrebno je omogućiti komunikaciju spremnika sa *host* poslužiteljem, tj. sa poslužiteljem na kojemu se Docker izvršava. To je moguće napraviti preko izlaganja (engl. *Expose*) *portova*. Unutar `ports` dijela konfiguracije za `mysql` servis se specificira vrijednost `3307:3306`. To znači da će se *port* `3306` unutar `mysql` spremnika izložiti te će se objaviti (engl. *Publish*) na *host* poslužitelju i biti će dostupan na *portu* `3307` na *host* poslužitelju. Na isti način će se objaviti *port* za `prestashop` servis koji specificira vrijednosti `8080:80` te `phpmyadmin` servis koji specificira vrijednosti `1235:80`. Sa tim procesom je omogućena komunikacija svih spremnika sa *host* poslužiteljem.

Također, osim komunikacije spremnika sa *host* poslužiteljem, potrebno je omogućiti i međusobnu komunikaciju između spremnika. Takva komunikacija se ostvaruje preko definiranja *networks* parametra. Jednostavno se upiše isto ime mreže za svaki pojedini servis i sa time Docker zna da mora kreirati mrežu tog imena u kojoj je moguća međusobna komunikacija spremnika.

Naposljetku, moguće je da jedan spremnik ovisi o drugome, tj. ne može funkcionirati bez njega. `Prestashop` i `phpmyadmin` servisi ovise o `mysql` servisu stoga je potrebno Docker Compose alatu naznačiti `depends_on` parametar. Time se naznačuje da se `prestashop` i `phpmyadmin` spremnici neće pokrenuti sve dok `mysql` spremnik nije spreman i u potpunosti pokrenut kako bi svi spremnici ispravno radili.

5.5. Izrada sigurnosne kopije

Prestashop aplikacija u Docker okruženju se pokreće preko `docker compose up` naredbe. Izvršavanjem te naredbe izvršavaju se sljedeći koraci:

- Docker pronalazi `docker-compose.yml` datoteku u trenutnome radnom direktoriju i čita njen sadržaj i upute
- Kreiraju se volumeni koji će spremnici koristiti na *host* poslužitelju
- Kreira se mreža u kojoj će spremnici moći međusobno komunicirati
- Docker traži definirane *image* datoteke na lokalnom poslužitelju te ako ne postoje, preuzima ih sa Docker Hub repozitorija
- Po uputama *image* datoteka, Docker kreira i pokreće navedene Docker spremnike

Nakon navedenih koraka spremnici su aktivni te je aplikacija spremna za rad i može joj se pristupiti sa *host* računala tako da se otvori poveznica `http://localhost:8080` u web pregledniku. Kreiranje sigurnosne kopije se sadrži od dva koraka:

1. Izrada sigurnosne kopije direktorija i datoteka aplikacije i konfiguracija
2. Logička sigurnosna kopija MySQL baze podataka pomoću `mysqldump` alata

5.5.1 Izrada sigurnosne kopije datoteka

Prethodno, u poglavlju 5.3 je već opisano da se za podatke koje želimo trajno sačuvati kreiraju Docker imenovani volumeni. Također je objašnjeno kako su ti podaci trajno dostupni na *host* poslužitelju. Jedna od karakteristika imenovanih volumena je da Docker u potpunosti upravlja sa njima. To je važno jer to znači da korisnik nema izravan pristup podacima na *host* poslužitelju. Posljedično, korisnik ne može izravno izraditi sigurnosnu kopiju tih podataka preko operativnog sustava *host* poslužitelja već mora koristiti Docker.

Kako bi korisnik izradio sigurnosnu kopiju preko Dockera, koristi se Docker *bind mount* i privremeni spremnik za ekstrakciju podataka iz volumena (Slika 23). Za izvršavanje tog procesa koristi se *full-backup.sh* skripta (Ispis 14).

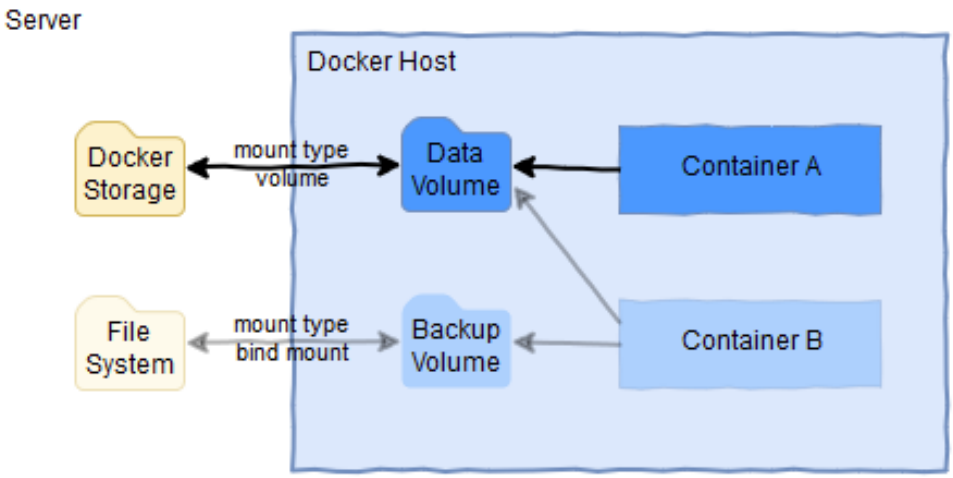
```
#! /bin/bash
docker run --rm \
--volumes-from prestashop-app \
--volumes-from prestashop-db \
--volumes-from phpmyadmin \
-v $(pwd)/configs-backup:/backup \
-v $(pwd)/prestashop-app-backup:/prestashop-files-backup \
ubuntu bash -c "tar cvf /prestashop-files-backup/prestashop-app.tar
/var/www/html && tar cvf /backup/mysql-cfng.tar /etc/mysql && tar cvf
/backup/apache2-cfng.tar /etc/apache2 && tar cvf /backup/php-cfng.tar
/etc/php && tar cvf /backup/pma-cfng.tar /etc/phpmyadmin"
rsync -av ./configs-backup /media/$USER/external_drive/
rsync -av ./prestashop-app-backup /media/$USER/external_drive/
```

Ispis 14: `full-backup.sh` skripta

Pokretanje skripte izvršava korake:

- Kreira se i pokreće privremeni spremnik prema uputama *image* datoteke `ubuntu` (ako ne postoji lokalno onda se dohvaća sa Docker Hub repozitorija).
- Direktoriji iz imenovanih volumena iz `prestashop-app`, `prestashop-db` i `phpmyadmin` spremnika se montiraju u privremeni spremnik na odgovarajuće putanje.
- Kreira se *bind mount*, tj prostor za pohranu čija je putanja `$(pwd)/prestashop-app-backup` na *host* poslužitelju. Taj direktorij će biti dostupan unutar spremnika na putanji `/backup`.
- Kreira se još jedan *bind mount* putanje `$(pwd)/prestashop-app-backup` na poslužitelju te će biti dostupan u spremniku na putanji `/prestashop-files-backup`.
- Kreira se i pokreće spremnik te se izvršava niz naredbi koje će pristupiti podacima u imenovanim volumenima i spremiti te podatke u arhive.

- Spremnik se gasi i uništava jer više nije potreban (`--rm`)
- Arhive se spremaju na sigurnu lokaciju pomoću `rsync` alata



Slika 23: Izrada sigurnosne kopije pomoću volumena i *bind mount* [17]

Činjenicom da se arhive specifično kreiraju u direktorijima koji su montirani unutar spremnika sa *host* poslužitelja pomoću *bind mount*, te arhive će biti dostupne na poslužitelju nakon što su kreirane. Tada korisnik ima izravan pristup arhiviranim podacima i može ih kopirati na sigurnu lokaciju te je sa time proces izrade sigurnosne kopije gotov.

5.5.2 Logička sigurnosna kopija podataka pomoću mysqldump alata

U poglavlju 2.4 je objašnjeno zašto se koristi `mysqldump` alat umjesto obične fizičke kopije datoteka za bazu podataka koja aktivno radi. Sukladno tome, unutar Docker platforme sigurnosna kopija baze podataka se neće raditi kao u poglavlju 5.5.1 već će se obavljati preko preporučenog `mysqldump` alata.

Sigurnosna kopija podignute (engl. *Live*) baze podataka će se izvršiti preko skripte imena `mysqldump-backup.sh` (Ispis 15).

```
sudo docker exec prestashop-db \
sh -c 'exec mysqldump prestashopdb -uroot -p"$MYSQL_ROOT_PASSWORD"' >
./prestashop-db-dump.sql
rsync -av ./prestashop-db-dump.sql /media/external_drive/
```

Ispis 15: `mysqldump-backup.sh`

Pomoću `mysqldump-backup.sh` skripte se unutar `prestashop-db` spremnika izvrši naredba `mysqldump` koja će kreirati *dump* datoteku, tj. logičku sigurnosnu kopiju `prestashopdb` baze podataka. Kreirana *dump* datoteka će se spremiti na *host* poslužitelj u trenutnom radnom direktoriju te će se pomoću `rsync` alata kopirati na sigurnu lokaciju. Naredbi, tj. `mysql` klijentu se pruža lozinka preko varijable okoline `MYSQL_ROOT_PASSWORD` koja je postavljena u `docker-compose.yml` datoteci (Ispis 13).

5.5.3 Automatizacija sigurnosne kopije pomoću cron alata

U poglavlju 2.7 je preporučeno raditi redovne sigurnosne kopije pomoću cron alata. Stoga će se isti princip primijeniti i za Docker platformu.

Činjenicom da su izrađene skripte za kreiranje sigurnosne kopije, jedino što je potrebno je izvršavati te skripte u redovnim intervalima. Kao što je već opisano, za potrebe Prestashop aplikacije dovoljno je raditi sigurnosnu kopiju jednom dnevno u 03:00 ujutro. Stoga će se napraviti dva cron posla u `crontab` konfiguracijskoj datoteci:

1. Cron posao za izvršavanje *`full-backup.sh`* skripte.

```
0 3 * * * /backup-scripts/full-backup.sh
```

2. Cron posao za izvršavanje `mysqldump-backup.sh` skripte.

```
0 3 * * * /backup-scripts/mysqldump-backup.sh
```

5.5.4 RAID1 zrcaljenje

U poglavlju 2.8 je objašnjeno da se koristi RAID1 konfiguracija diskova da bi se postigla dodatna sigurnost podataka. Kako se sigurnosne kopije spremaju i jer Docker volumeni „žive“ na *host* poslužitelju, ima smisla da se RAID1 konfiguracija napravi za disk na kojem je operativni sustav *host* poslužitelja. Postupak izrade RAID1 konfiguracije će biti identičan kao u poglavlju 3.5.2 jer se RAID1 konfiguracija postavlja za diskove *host* poslužitelja.

5.6. Obnova poslužitelja, Docker alata i podataka

Proces obnove poslužitelja će se sastojati od sljedećih koraka:

1. Instalacija operativnog sustava
2. Instalacija Docker i Docker Compose
3. Kopiranje sigurnosne kopije podataka na poslužitelj
4. Izvlačenje podataka iz arhiva u *host* direktorije imenovanih volumena
5. Uvoz podataka iz *dump* datoteke u bazu podataka

Instalacija operativnog sustava će biti klasična instalacija kao i za obični fizički poslužitelj. Razlika u obnovi poslužitelja dolazi tek nakon instalacije operativnog sustava. Velika prednost korištenja Docker platforme je nepotrebnost instalacije alata i softvera izravno na *host* poslužitelju potrebnih za rad Prestashop aplikacije. Softver potreban za rad aplikacije će se instalirati unutar Docker spremnika stoga je na poslužitelju potrebno instalirati samo Docker platformu i Docker Compose alat.

Instalacija Docker platforme na Debian 11 operacijskom sustavu preko Docker repozitorija je vrlo jednostavna. Potrebno je samo namjestiti (engl. *Set up*) Docker repozitorij i nakon toga instalirati Docker *engine* i Docker Compose. To se obavlja na sljedeći način:

1. Ažuriranje indeksa APT paketa i instaliranje potrebnih paketa kako bi omogućili APT alatu korištenje repozitorija preko HTTPS-a (*Hypertext Transfer Protocol Secure*) (Ispis 16).

```
sudo apt-get update && \
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

Ispis 16: Naredbe za ažuriranje indeksa APT paketa i instalacija potrebnih paketa

2. Dodavanje službenog Docker GPG (*GNU Privacy Guard*) ključa (Ispis 17).

```
sudo mkdir -p /etc/apt/keyrings && \
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --
dearmor -o /etc/apt/keyrings/docker.gpg
```

Ispis 17: Naredbe za dodavanje ključa

3. Namještanje Docker repozitorija (Ispis 18).

```
echo "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list
> /dev/null
```

Ispis 18: Namještanje repozitorija

4. Instalacija Docker *engine* i Docker Compose (Ispis 19).

```
sudo apt-get update && \
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-
compose-plugin
```

Ispis 19: Docker instalacija

Kopiranje sigurnosne kopije na poslužitelj se može postići pomoću `rsync` alata u željeni direktorij (Ispis 20):

```
sudo rsync -av /media/external_drive/docker-compose.yml
/home/$USER/docker-prestashop
sudo rsync -av /media/external_drive/configs-backup /home/$USER/docker-
prestashop
sudo rsync -av /media/external_drive/prestashop-app-backup
/home/$USER/docker-prestashop
```

Ispis 20: Naredbe za kopiranje sigurnosne kopije na poslužitelj

Prije izvlačenja podataka iz arhiva u direktorije Docker volumena, potrebno je prvo pokrenuti Docker spremnike. To se vrši preko naredbe `docker compose up` unutar radnog direktorija u kojoj se nalazi `docker-compose.yml` datoteka. Kada su spremnici pokrenuti i aktivni, mogu se pokrenuti skripte za obnovu podataka. Potpuna obnova podataka koja uključuje izvlačenje podataka iz arhiva i obnove baze podataka preko `mysqldump` alata se čini preko `complete-restore.sh` skripte (Ispis 21).

```
sudo docker run --rm \
--volumes-from prestashop-app \
```

```
--volumes-from prestashop-db \
--volumes-from phpmyadmin \
-v $(pwd)/../configs-backup:/config-files-backup \
-v $(pwd)/../prestashop-app-backup:/prestashop-files-backup \
ubuntu bash -c "cd /var/www/html && rm -rf * && tar xvf /prestashop-
files-backup/prestashop-app.tar --strip 1 && cd /etc/mysql && tar xvf
/config-files-backup/mysql-cfng.tar --strip=2 && cd /etc/apache2 && tar
xvf /config-files-backup/apache2-cfng.tar --strip=2 && cd /etc/php && tar
xvf /config-files-backup/php-cfng.tar --strip=2 && cd /etc/phpmyadmin &&
tar xvf /config-files-backup/pma-cfng.tar --strip=2"
sudo docker exec -i prestashop-db sh -c 'exec mysql -uroot -
p"$MYSQL_ROOT_PASSWORD" prestashopdb' < ../prestashop-db-dump.sql
```

Ispis 21: `complete-restore.sh` skripta

Slično kao i kod kreiranja sigurnosne kopije, korisnik nema izravan pristup direktorijima koji su montirani u spremnik sa *host* poslužitelja. Iz tog razloga je potrebno ponovno pokrenuti privremeni Docker spremnik pomoću kojeg će se izvršiti obnova podataka. Prvo, Skripta unutar privremenog Docker spremnika mijenja trenutni radni direktorij u odgovarajuće putanje i u tim direktorijima izvlači podatke iz arhiva koje pruža *bind mount*. Nakon izvlačenja datoteka iz arhiva u direktorije, datoteke će biti dostupne u spremnicima `prestashop-app`, `prestashop-db` i `phpmyadmin` jer su ti direktoriji montirani preko imenovanih volumena.

Naposljetku, skripta izvršava obnovu baze podataka pomoću `docker exec` naredbe i `mysqldump` alata. Sa tim korakom proces obnove poslužitelja je gotov i aplikacija je spremna za rad sa aktualnim podacima.

6. Usporedba

U tablici (Tablica 1) je prikazana usporedba modela za različite parametre prilikom instalacije sustava, izrade sigurnosne kopije i obnove poslužitelja.

	Fizički poslužitelj	Virtualni Stroj	Docker
Kompleksnost inicijalnog postavljanja i instalacije sustava	Mala	Mala/Osrednja	Velika
Vrijeme izrade sigurnosne kopije	Kratko	Dugo	Osrednje
Kompleksnost izrade sigurnosne kopije	Mala	Osrednja	Velika
Kompleksnost obnove	Velika	Mala	Mala
Vrijeme potrebno za obnovu	Dugo	Kratko	Kratko/Osrednje

Tablica 1: Usporedba modela

Sva tri modela imaju jednostavnu automatizaciju izrade sigurnosne kopije preko cron posla, tj. izvršavanje skripti u redovnim intervalima. Osim toga, sve tri metode mogu iskoristiti prednosti korištenja RAID1 konfiguracije diskova jer su sva tri sustava esencijalno na disku *host* poslužitelja.

Također, u prethodnim poglavljima su opisani potencijalni problemi koji se mogu dogoditi u različitim modelima. Imajući sve to na umu, očigledno je da nema savršenog modela koji je najbolji u svim aspektima.

7. Zaključak

Izrada sigurnosne kopije jedna je od ključnih stavki poslovanja u koju bi se trebalo utrošiti vremena i resursa da podaci poslovanja ostanu zaštićeni. Na jednom sustavu može postojati mnogo podataka i različitih komponenti za koje se treba izraditi sigurnosna kopija kako se podaci ne bi izgubili.

Postoji mnogo načina izrade sigurnosne kopije za različite komponente sustava. U radu su pokazane samo neke metode izrade kopije za modele fizičkog poslužitelja, virtualnog stroja i Docker platforme. Moglo se vidjeti da je postavljanje okruženja, instalacija alata, izrada sigurnosne kopije i oporavak različit za svaki od opisanih modela iako svaki model ima istu svrhu.

Pomoću cron alata se može vršiti automatizacija izrade sigurnosne kopije preko cron poslova koji se izvršavaju u redovnim intervalima. Nadalje, pokazano je kako RAID1 konfiguracija diskova može dodatno poboljšati zaštitu podataka tako da se podaci sa jednog diska zrcale na drugi disk. RAID1 konfiguraciju je moguće napraviti na dva načina: Preko hardvera ili softvera.

Kako bi sustav bio efikasan u jednom segmentu vrlo često mora raditi kompromise u drugim segmentima sustava. Odabir najpogodnijeg i najefikasnijeg modela će ovisiti o sposobnostima korisnika (administratora), vremenu koje ima na raspolaganju, budžetu, dostupnom hardveru i slično. Može se zaključiti da je do korisnika da odluči koji model bi njemu bio prikladan za administraciju i rad poslužitelja.

8. Literatura

[1] „About Debian“, https://www.debian.org/intro/about (15.06.2022)

[2] „What is Apache“, https://www.greengeeks.com/blog/what-is-apache (15.06.2022)

[3] „Usage of web servers“, https://w3techs.com/technologies/overview/web_server (15.06.2022)

[4] „What is PHP“, https://www.php.net/manual/en/intro-whatis.php (17.06.2022)

[5] „What is MySQL“, https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html (18.06.2022)

[6] „Backup and recovery types“, https://dev.mysql.com/doc/refman/8.0/en/backup-types.html (18.06.2022)

[7] „About QEMU“, https://www.qemu.org/docs/master/about/index.html (23.06.2022)

[8] „What is KVM“, https://www.redhat.com/en/topics/virtualization/what-is-KVM (23.06.2022)

[9] „Docker overview“, https://docs.docker.com/get-started/overview/ (03.07.2022)

[10] „Use volumes“, https://docs.docker.com/storage/volumes/ (03.07.2022)

[11] „Cron job“, https://www.hostinger.com/tutorials/cron-job (05.07.2022)

[12] „RAID“, https://www.techtarget.com/searchstorage/definition/RAID (15.07.2022)

[13] „What is a bare-metal server“, https://www.rackspace.com/library/what-is-a-bare-metal-server (16.07.2022)

[14] „Rsync“, https://linux.die.net/man/1/rsync (16.07.2022)

[15] „Activating the Intel VT Virtualization Feature“, https://www.thomas-krenn.com/en/wiki/Activating_the_Intel_VT_Virtualization_Feature (22.07.2022)

[16] „Virsh“, https://www.libvirt.org/manpages/virsh.html (22.07.2022)

[17] „Backup for Docker volumes“, https://scorban.de/2018/02/06/auto-backup-fuer-docker-volumes/ (24.07.2022)

POPIS OZNAKA I KRATICA

APT - Advanced Package Tool

LAMP - Linux, Apache, MySQL, PHP

HTTP - Hypertext Transfer Protocol

PHP - PHP: Hypertext Preprocessor

SQL - Structured Query Language

QEMU - Quick Emulator

KVM - Kernel-based Virtual Machine

RAID - Redundant Array of Independent Disks

BIOS - Basic Input/Output System

API - Application Programming Interface

ISO - Image Standard Optical

XML - Extensible Markup Language

YAML – YAML Ain't Markup Language

GPG - GNU Privacy Guard

IME I PREZIME: Roko Švenjak
Adresa: Put Vrela 60, 23000 Zadar
OIB: 41475630157

IZJAVA O AUTENTIČNOSTI ZAVRŠNOG RADA

Izjavljujem da sam završni rad pod nazivom

Planovi obnove web poslužitelja

izradio/la samostalno.

Svi dijelovi rada rezultat su isključivo mojega vlastitog rada i temelje se na mojim istraživanjima. Dijelovi rada koji su citirani iz različitih izvora jasno su označeni kao takvi te navedeni u fusnoti i literaturi.

Split, 22.08.2022

Ime i Prezime

ROKO ŠVENJAK