TRANSMISI : JURNAL ILMIAH TEKNIK ELEKTRO, 24, (2), APRIL 2022
p-ISSN 1411-0814 e-ISSN 2407-6422

PENGEMBANGAN DAN IMPLEMENTASI DOCKER UNTUK MEMAKSIMALKAN UTILITAS SERVER UNIVERSITAS PADA MASA COVID-19

Rama Aria Megantara, Farrikh Alzami[*)], Ricardus Anggi Pramunendar dan Dwi Puji Prabowo

Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia
Jl. Imam Bonjol 207 Semarang

[*)]*E-mail: alzami@dsn.dinus.ac.id*

Abstrak

Covid-19 membuat seluruh kegiatan produktif manusia terdisrupsi, termasuk pada pendidikan. Kegiatan belajar yang seharusnya tatap muka kemudian berganti menjadi daring, memaksa universitas untuk membelanjakan anggaran tidak terduga untuk mendukung infrastruktur pembelajaran secara daring terutama *server* dan sistem pendidikan daring yang tangguh. Kegiatan mahasiswa praktek untuk penggunaan alat maupun melakukan pemprograman yang membutuhkan sumber daya yang cukup besar. Penelitian ini bertujuan mengembangkan optimalisasi utilitas server universitas untuk membantu mahasiswa menggunakan sumber daya Perguruan Tinggi tanpa mengeluarkan biaya tambahan seperti pembelian komputer baru maupun penambahan biaya listrik. Pengembangan ini berfokus pada pendekatan pemrograman menggunakan Linux dan docker. Dari hasil kuesioner, didapatkan bahwa server docker yang dibangun telah membantu 40 partisipan dalam menjalankan kegiatan belajar mengajar dan penelitian.

Kata kunci: Docker, Cloud Computing, Server, covid-19

Abstract

Covid-19 has disrupted all human productive activities, including education. Learning activities that were supposed to be face-to-face then turned online, forcing universities to spend unexpected budgets to support online learning infrastructure, especially servers and a robust online education system. The problem that the researcher raises is the students' practical activities for the use of tools and for programming which requires considerable resources. By focusing on programming problems, researchers use a Linux and docker approach to help students use university resources without incurring additional costs such as purchasing a new computer or adding electricity costs. From the results of the questionnaire, it was found that the Docker Server that was built had helped 40 participants in carrying out teaching and learning activities and research.

Keywords: Docker, Cloud Computing, Server, covid-19

1. Pendahuluan

Covid-19 membuat seluruh kegiatan produktif manusia terdisrupsi, tak terkecuali pada Pendidikan. Covid-19 memaksa dunia Pendidikan yang mulanya belajar tatap muka menjadi pembelajaran secara virtual atau daring[1]. Banyak Lembaga Pendidikan gagap akan kondisi ini dan harus mengeluarkan kebijakan agar proses belajar mengajar dapat terus berlangsung, tak terkecuali Perguruan Tinggi [2]. Dibandingkan dengan Lembaga Pendidikan yang lain, Perguruan tinggi tidak sampai mati suri dalam kegiatan belajar mengajarnya, karena banyak mahasiswa dan dosen yang sudah menggunakan ponsel pintar dan computer, serta terbiasa menggunakan internet dibandingkan dengan peserta didik dibawahnya (SD, SMP, atau SMA). Namun demikian, kegiatan belajar yang seharusnya tatap muka kemudian berganti menjadi daring, memaksa universitas untuk membelanjakan anggaran tidak terduga untuk mendukung infrastruktur pembelajaran secara daring terutama server dan system Pendidikan daring yang tangguh [3]. Ada beberapa solusi yang bisa digunakan dalam mensiasati kegiatan pembelajaran secara daring, antara lain: menggunakan learning management system sebagai system informasi pusat pembelajaran; kemudian aplikasi konferensi seperti zoom, google meet, teams atau webex untuk menghadirkan pembelajaran tatap muka virtual. Namun, hal yang belum teratasi adalah kegiatan mahasiswa praktek untuk penggunaan alat maupun melakukan pemprograman yang membutuhkan sumber daya yang cukup besar. Disini penulis mencoba memecahkan masalah pada bagian mahasiswa yang terkendala sumber daya dalam melakukan pemprograman. Alasan utama adalah: 1) mahasiswa tidak mempunyai computer yang cukup untuk melakukan pemprograman

karena mahasiswa terbiasa melakukan pemprograman didalam lab computer; 2) Perguruan Tinggi tidak mungkin mengonlinekan lab computer nya sehingga mahasiswa bisa meremote computer lab dari rumah karena keterbatasan bandwith dan ruangan harus selalu dijaga oleh laboran, sedangkan kondisi covid memaksa perguruan tinggi untuk menerapkan bekerja dari rumah; 3) dosen harus mampu mengecek dan mengevaluasi kegiatan pemprograman mahasiswa, baik dari sisi intensitas penggunaan sumber daya dan hasil pembelajaran menggunakan pemprograman tersebut.

Penyelesaian masalah tersebut dapat menggunakan konsep virtualisasi berbasis mesin virtual (*virtual machine*, VM). Suatu server dapat diisi beberapa server virtual. Sebagaimana server riil, server virtual harus mempunyai system operasi ((*operating system* atau OS) dan berbagai perangkat lunak lain agar layanan *web hosting* dapat berjalan. Dengan adanya server virtual, setiap aplikasi berjalan pada mesinnya masing-masing dimana file pustaka, *interpreter*, server database setiap server disesuaikan dengan kebutuhan dari aplikasi[4]. Salah satu usaha untuk mendukung virtualisasi server adalah menggunakan docker. Docker *container* merupakan salah satu aplikasi yang dapat mengelola banyak aplikasi[5], kemudian Docker *Container* sangat cocok untuk desain arsitektur sistem dengan pendekatan *microservice*, karena masing-masing layanan memiliki lingkungan yang terisolasi namun tetap dapat berkomunikasi satu sama lain serta dapat meningkatkan efektivitas dalam penggunaan sumber daya *central processing unit* (CPU) dan memori pada server [6]. Penelitian yang dilakukan Potdar dkk, menyebutkan bahwa dari sisi kinerja CPU, *memory throughput*, Disk I/O, *load test* dan pengukuran kecepatan operasi, menghasilkan kesimpulan bahwa docker lebih baik dibandingkan dengan VM [7]. Docker juga digunakan di banyak lini bidang, antara lain: pemodelan pendukung keputusan untuk peneliti lingkungan [8], container deep learning untuk segmentasi citra biomedis[9], manajemen software pada perusahaan dan manufaktur [10]

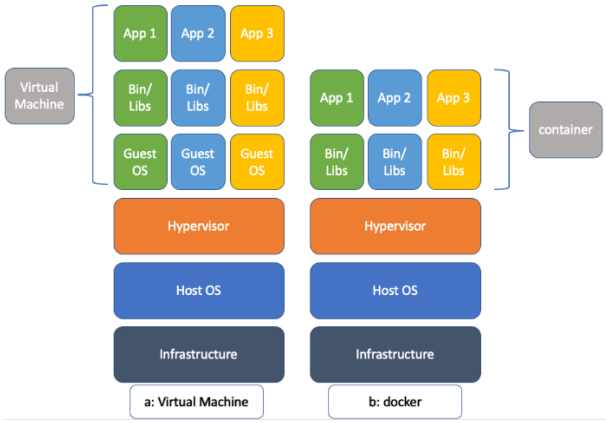
Pada prakteknya, docker container disimpan dalam bentuk image. Dengan bentuk image, maka kelebihan lain menggunakan docker adalah dengan mengandalkan modularitasnya, sebuah image dapat dibagikan dan disertakan dalam image lain untuk menyederhanakan cara membangun dan mengemas perangkat lunak baru (building and packaging new software)[11] serta mempermudah mengirim aplikasi tradisional menjadi aplikasi cloud / serverless [12]. Untuk manajemen *container*, portainer digunakan untuk kemudahan pembuatan images kepada pengguna dengan spesifikasi yang berbeda-beda sesuai dengan penelitian yang dilakukan oleh Balatamoghna dkk [13]. Kekurangan pada docker adalah konsumsi energy yang dibutuhkan cukup tinggi karena kebutuhan I/O system calls cukup besar dibandingkan dengan server yang tidak menggunakan service docker[14].

Karena tiap mahasiswa mempunyai spesifikasi kebutuhan pemprograman yang beragam, maka penelitian ini bertujuan mengembangkan server docker yang spesifik agar para mahasiswa dapat menggunakan infrastruktur pemprograman sesuai dengan kebutuhan Perguruan Tinggi berdasarkan kebutuhan modularitasnya. Kemudian, untuk mengetahui efektivitas docker yang dikembangkan, dilakukan survei terhadap pengguna.

2. Metode Penelitian

2.1. Tahap Perencanaan

Docker dipilih pada penelitian ini dibandingkan dengan virtual machine berdasarkan pada perangkat keras yang tersedia di Perguruan Tinggi. Perangkat keras yang tersedia pada Perguruan Tinggi cukup terbatas dari sisi Penyimpanan dan RAM. Virtual machine bersandar pada Guest OS, maka berakibat penggunaan sumber daya server menjadi berlebih; sedangkan pada docker, berbagi bersama OS sehingga lebih ringan dan lebih tanggap jika digunakan [15].



Gambar 1. Perbedaan Virtual Machine dan Docker

Gambar 1 menunjukkan bahwa berbagi sistem operasi host di antara container membuatnya sangat ringan dan membantu proses *boot* hanya dalam beberapa detik. Oleh karena itu, biaya untuk mengelola sistem kontainer sangat rendah dibandingkan dengan mesin virtual. Docker container cocok untuk situasi menjalankan beberapa aplikasi melalui satu kernel sistem operasi (dalam hal ini linux). Tetapi jika terdapat aplikasi atau server yang perlu dijalankan pada sistem operasi yang berbeda, maka mesin virtual dapat menjadi alternatif.

Karena pemprograman yang umumnya dipakai di Perguruan Tinggi tempat peneliti bekerja menggunakan open source, maka peneliti menggunakan docker dengan Linux sebagai Host OS. Hal ini terkait dengan kemudahan dalam instalasi library pendukung pemprograman dibandingkan dengan windows, lalu karena linux merupakan OS yang gratis, serta mendukung untuk

kegiatan text based only, dimana para mahasiswa tidak membutuhkan bandwith yang tinggi dalam mengoperasikan docker mereka dan hanya perlu fokus pada pemprogramannya.

Karena server diakses oleh mahasiswa secara remote diluar kampus dan menggunakan internet, maka keamanan menjadi hal utama. Disini peneliti menggunakan CentOS karena OS tersebut sudah menerapkan Security-Enhanced Linux (SELinux). CentOS juga mempunyai package management yang lebih lengkap untuk kegiatan produksi dalam server. Dalam artian, jika kita menggunakan server sebagai pelayanan realtime kepada pengguna, maka CentOS mampu melayani dengan performa maksimal.

Dan yang terakhir, CentOS mendukung banyak platform management, seperti cPanel, InterWorx, WebMin, DirectAdmin, Spacewalk, CWP, Plesk, ISPConfig, Virtualmin, Vesta CP yang memudahkan peneliti dalam mengatur situs yang diakses oleh mahasiswa untuk menggunakan docker.

Untuk partisi harddisk, digunakan konfigurasi seperti pada Gambar 2. Partisi sebesar 1 GB untuk *boot* sebagai penyimpanan kernel karena jika ada kernel baru dan tidak cocok, maka lebih mudah dalam melakukan *rollback* ke kernel sebelumnya.

```
[root@server-pdik ~]# df -h
Filesystem              Size  Used Avail Use% Mounted on
devtmpfs                126G     0  126G   0% /dev
tmpfs                   126G     0  126G   0% /dev/shm
tmpfs                   126G  4.1G  122G   4% /run
tmpfs                   126G     0  126G   0% /sys/fs/cgroup
/dev/mapper/centos-root  3.7T  471G  3.2T  13% /
/dev/sdb1               3.6T   90M  3.4T   1% /root/DATA
/dev/sda2              1014M  184M  831M  19% /boot
```

Gambar 2. Konfigurasi Harddisk

2.2. Tahap Setting NVIDIA-CUDA

GPU digunakan karena banyak mahasiswa menggunakan pemprograman deep learning yang menggunakan CUDA sebagai base driver nya. Untuk instalasi CUDA pada CentOS cukup mudah dengan terlebih dahulu menginstall beberapa library sebagai berikut:

Install Group Development dari driver
yum groupinstall "Development Tools"
Kemudian install kernel agar support menjalankan GPU
yum install kernel-devel epel-release
Selanjutnya install DKMS (Dynamic Kernel Module Support) agar ketika sistem mengupdate kernel otomatis akan menambahkan modul GPU.
yum install dkms
Kemudian edit bagian GRUB untuk mematikan “nouveau” dengan menambahkan script ini
“nouveau.modeset=0” pada “/etc/default/grub”.
Kemudian update grub dengan command :
grub2-mkconfig -o /boot/efi/EFI/centos/grub.cfg
Kemudian lakukan instalasi driver dengan command :
bash NVIDIA-Linux-x86_64-*

Perlu digaris bawahi, Nouveau adalah driver opensource yang sudah tertanam pada sistem linux. Untuk menggunakan driver dari web resmi maka harus didisable dahulu agar instalasi berhasil.

2.3. Tahap Setting CUDA ToolKit

CUDA Toolkit mencakup perpustakaan yang dipercepat oleh GPU, kompiler, alat pengembangan, dan runtime CUDA. Toolkit ini sangat diperlukan jika akan menggunakan GPU untuk melakukan pemprosesan pada deep learning.

Mengunduh repo CUDA Toolkit
wget cuda.rpm
Kemudian lanjutkan untuk menginstall dengan command :
rpm -i cuda-repo-*.rpm
Selanjutnya install CUDA dengan command :
yum install cuda
Kemudian tambahan export variable cuda agar dapat berjalan otomatis.
export PATH=/usr/local/cuda/bin:$PATH export LD_LIBRARY_PATH=/usr/local/cuda/lib64:$LD_LIBRARY_PATH export LD_LIBRARY_PATH=/usr/local/cuda/lib64::/usr/local/lib:/usr/local/lib
Kemudian untuk memastikan, lakukan command dibawah ini:
nvcc --version # nvidia-smi

```
[root@server-pdik ~]# nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2020 NVIDIA Corporation
Built on Mon_Oct_12_20:09:46_PDT_2020
Cuda compilation tools, release 11.1, V11.1.105
Build cuda_11.1.TC455_06.29190527_0
[root@server-pdik ~]# nvidia-smi
Tue Feb 16 21:35:12 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 455.45.01    Driver Version: 455.45.01    CUDA Version: 11.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce RTX 3070    Off  | 00000000:18:00.0 Off |                  N/A |
| 77%   67C    P2   264W / 270W |    663MiB /  7982MiB |    100%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
```

Gambar 3. hasil instalasi Cuda Toolkit berhasil

Gambar 3 menunjukkan bahwa hasil instalasi cuda toolkit berhasil, sehingga memudahkan mahasiswa dalam melakukan pemprograman yang berkaitan dengan deep learning

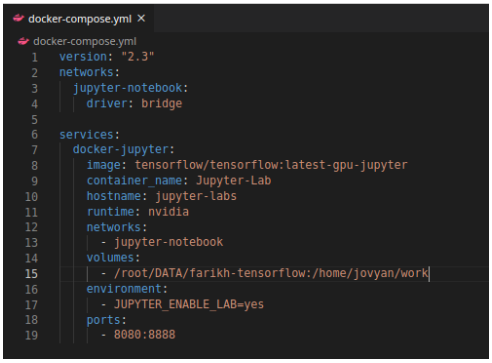
2.4. Tahap Instalasi Docker pada CentOS

Instalasi docker pada centos 7 sebagai sistem yang akan menampung berbagai service / layanan yang digunakan seperti jupyter notebook, portainer, nginx, dan service pendukung lainnya.

Tambahkan repo docker pada Centos
sudo yum install -y yum-utils # sudo yum-config-manager \ --add-repo \ *Instalasi ke web docker CE*
Kemudian install docker
sudo yum install docker-ce docker-ce-cli containerd.io
Setelah terinstall, jalankan docker.
sudo systemctl start docker
Kemudian tambahkan user ke group docker agar dapat dijalankan tanpa menggunakan root permission.
usermod -aG docker {user}
Kemudian install docker-compose untuk kebutuhan ketika membuat container menggunakan script. Kelebihan membuat script adalah ketika server atau beberapa container crash dapat di ulang membuat container karena adanya script docker. Script ini berbasis YAML, yang berisikan configurasi container. Install docker-compose dengan cara :
sudo curl -L "*situs ke github docker compose*" #(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
Kemudian ubah permisi dari script docker-compose agar dapat di eksekusi oleh mesin.
chmod +x /usr/local/bin/docker-compose
Agar dapat dipanggil dalam command line atau terminal, buat symbolic link ke /usr/bin/ dengan command :
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
Ketika semua sudah terinstalasi, cek dengan cara :
docker -v && docker-compose -v
Selanjutnya install docker runtime agar container dapat menjalankan CUDA Toolkit. Tambahkan repo dibawah ini
distribution=$(. /etc/os-release;echo IDVERSION_ID) # curl -s -L "*situs ke nvidia github io*"
Kemudian install dengan perintah :
yum install -y nvidia-container-runtime
Kemudian tambahkan runtime ke docker daemon.json pada "/etc/docker/daemon.json" seperti gambar dibawah ini.

"runtimes": {
"nvidia": {
"path": "/usr/bin/nvidia-container-runtime", "runtimeArgs":[]
}
}

Kemudian restart docker dan check dengan perintah dibawah ini.
systemctl restart docker && docker info|grep -i runtime
Lakukan testing menggunakan container nvidia/cuda:10.0-base
docker run -rm --runtime nvidia nvidia/cuda:10.0-base nvidia-smi



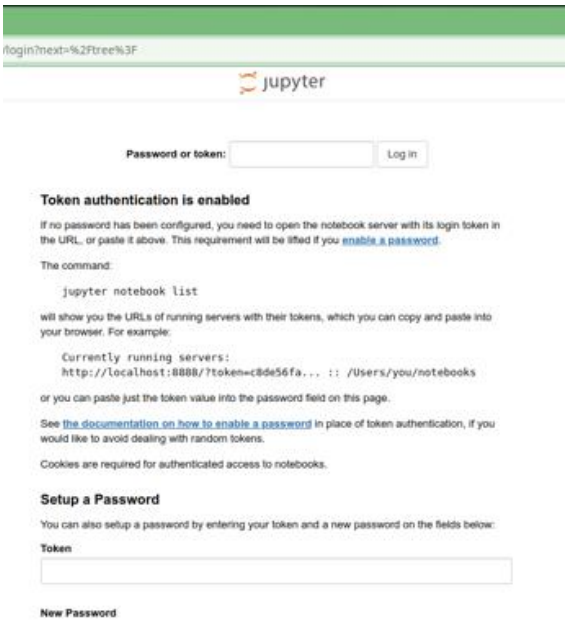
Gambar 4. Setting docker-compose.yml

Setelah berjalan, kemudian dibuat docker-compose.yml yang berisi configurasi untuk container jupyter-notebook, image docker disini menggunakan versi tensorflow karena didalam container sudah built'in nvidia-smi dan hanya melakukan penambahan runtime seperti pada gambar 4.

Kemudian docker-compose.yml dijalankan menggunakan command :

docker-compose up -d

Selanjutnya akses container dengan alamat web maka akan tampil dashboard awal jupyter-notebook seperti gambar 5

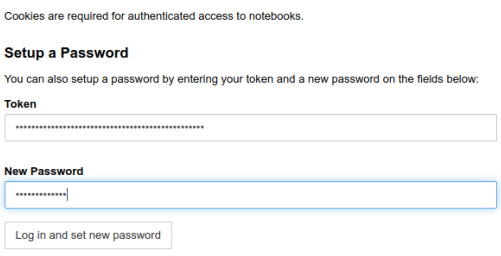


Gambar 5. Instalasi jupyter-notebook berhasil

Konfigurasi password dilakukan dengan menyalin kode token. Pada docker untuk mendapatkan token tersebut dari log dengan cara :

docker logs {nama_container}

Setelah dikonfigurasi maka akan dilakukan login dan dashboard utama sudah muncul seperti pada gambar 6.



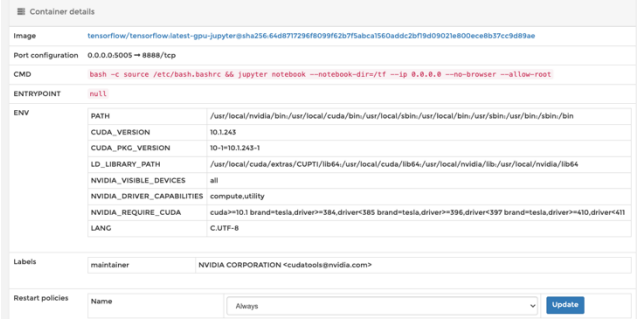
Gambar 6. Setting password jupyter-notebook

3. Hasil dan Analisis

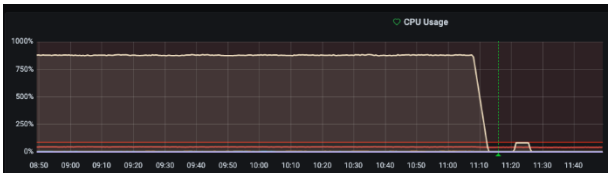
3.1. Manajemen docker

Untuk kemudahan manajemen container images, digunakan portainer. Portainer ini dalam bentuk GUI sehingga lebih mudah dibandingkan dengan text-based command prompt seperti ditampilkan pada gambar 7.

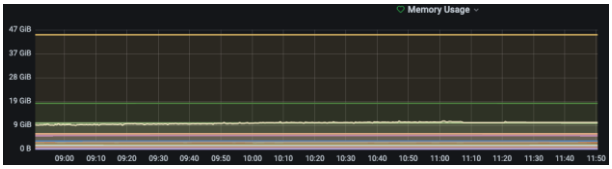
Selain itu, untuk mengetahui tingkat utilitas dari server yang menggunakan docker container, digunakan Grafana.



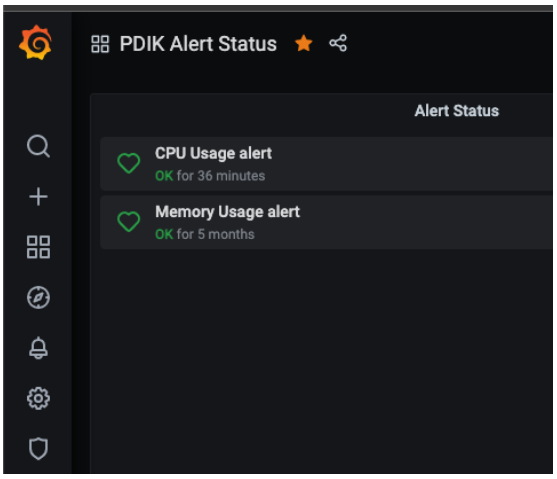
Gambar 7. Penggunaan portainer



Gambar 8. Status CPU Usage



Gambar 9. Status memory usage



Gambar 10. Alert pada Grafana

Pada Gambar 8 dan 9, status penggunaan GPU dan memori dapat dipantau. Dengan adanya informasi ini, dapat diketahui siapa pengguna (mahasiswa) yang menggunakan sumber daya paling banyak. Untuk kemudahan pemberian notifikasi, Grafana menyediakan widget alert yang dapat dilihat di cek oleh administrator, seperti ditunjukkan pada Gambar 10.

Dari hasil yang ditunjukkan pada Gambar 8-10, sistem telah dapat berjalan dan menunjukkan pengelolaan sumber daya infrastruktur server dengan baik.

3.2. Evaluasi penggun a

Setelah infrastruktur berhasil dipasang, peneliti meminta 40 partisipan untuk bergabung menggunakan infrastruktur server docker ini. Partisipan ini adalah mahasiswa doktoral yang sedang dalam kegiatan menyelesaikan disertasi dimana intensitas kebutuhan penggunaan server untuk pemprograman cukup tinggi. Setelah beberapa waktu menggunakan server docker ini, peneliti meminta 40 partisipan untuk mengisi kuesioner dengan pertanyaan sebagai berikut:

Tabel 1. Pertanyaan Kuesioner

Kode	Pertanyaan
Q1	Saya memiliki perangkat canggih yang mendukung penelitian saya
Q2	Saya Mendapatkan hak akses ke Server Docker dengan Mudah
Q3	Saya dapat menggunakan lebih dari dua docker image untuk kegiatan penelitian saya di Server Docker
Q4	Saya dapat mengakses server Docker melalui perangkat saya tanpa mengalami kendala
Q5	Saya percaya bahwa data penelitian saya aman di Server Docker karena Universitas mempunyai Tim IT yang handal
Q6	Saya terbantu dengan adanya Server Docker ini dalam penelitian saya
Q7	Saya merasa bahwa pengelola Server Docker sudah adil dalam memberikan porsi sumber daya ke setiap pengguna

Dengan menggunakan skala likert seperti tabel 2, peneliti mendapatkan hasil kuesioner pada tabel 3 sebagai berikut:

Tabel 2. Skala Likert yang digunakan

Kode	Keterangan
1	Sangat Tidak Setuju
2	Tidak Setuju
3	Netral
4	Setuju
5	Sangat Setuju

Tabel 3. Hasil Skor Likert

	Sangat tidak setuju	Tidak Setuju	Netral	Setuju	Sangat Setuju
Q1	0%	33%	63%	3%	3%
Q2	0%	0%	10%	10%	80%
Q3	0%	0%	8%	78%	15%
Q4	0%	0%	0%	10%	90%
Q5	0%	0%	0%	85%	15%
Q6	0%	0%	0%	25%	75%
Q7	0%	0%	0%	88%	13%

Dari pertanyaan Q1 ‘Saya memiliki perangkat canggih yang mendukung penelitian saya’, mayoritas menjawab **netral** (63%), karena banyak partisipan sudah mempersiapkan diri dengan membekali komputer canggih, meskipun demikian partisipan tidak tahu apakah komputer mereka cukup sampai akhir disertasi nanti. Kemudian 33%

partisipan menjawab **tidak setuju** karena komputer yang mereka siapkan tidak mendukung untuk disertasi mereka.

Dari pertanyaan Q2 'Saya Mendapatkan hak akses ke Server Docker dengan Mudah', mayoritas menjawab **sangat setuju** (80%) karena partisipan cukup menghubungi coordinator Lab Server Docker dengan membawa surat persetujuan penggunaan sumber daya oleh Kepala Program Studi.

Dari Pertanyaan Q3 'Saya dapat menggunakan lebih dari dua docker image untuk kegiatan penelitian saya di Server Docker', mayoritas menjawab **Setuju** (78%) karena partisipan juga menggunakan metode komparasi dalam penelitian, sehingga coordinator selalu memberikan izin untuk menambah kuota docker

Dari Pertanyaan Q4 'Saya dapat mengakses server Docker melalui perangkat saya tanpa mengalami kendala', mayoritas menjawab **sangat setuju** (90%) karena partisipan cukup login menggunakan internet, lalu mengerjakan tugas pemprograman, kemudian decompile selama beberapa waktu, setelah selesai, baru Kembali mengecek. Dari sini para partisipan tidak perlu membeli komputer lagi, dan hemat biaya listrik.

Dari Pertanyaan Q5 'Saya percaya bahwa data penelitian saya aman di Server Docker karena Universitas mempunyai Tim IT yang handal', mayoritas menjawab **setuju** (85%) karena para partisipan sudah dibekali pengetahuan tentang keamanan siber di Perguruan Tinggi tersebut.

Dari Pertanyaan Q6 'Saya terbantu dengan adanya Server Docker ini dalam penelitian saya', mayoritas menjawab **Sangat Setuju** (75%) karena mereka tinggal menggunakan docker yang sudah disediakan. Partisipan juga bisa menginstall apapun library yang diberikan. Namun, sebanyak 25% partisipan mengatakan setuju karena mereka tidak nyaman dengan pemprograman text based (browser based), yang mereka inginkan adalah dalam bentuk GUI.

Dari Pertanyaan Q7 'Saya merasa bahwa pengelola Server Docker sudah adil dalam memberikan porsi sumber daya ke setiap pengguna', mayoritas menjawab Setuju (88%) karena coordinator Lab selalu mengawasi server, sehingga jika ada partisipan yang menghabiskan resource server, maka coordinator Lab akan menghubungi dan memberi waktu untuk memperbaiki kesalahan. Jika tidak diindahkan, maka koordinator lab akan mematikan docker dari partisipan tersebut.

4. Kesimpulan

Disrupsi pada kegiatan belajar mengajar di masa pandemi menyebabkan gangguan aktivitas tatap muka serta terputusnya akses mahasiswa menggunakan Lab. Karena keterbatasan sumber daya dalam melakukan remote komputer lab, maka peneliti telah mengembangkan Server docker agar para mahasiswa dapat menggunakan sumber daya server dimanapun dan kapanpun dengan *fair usage policy*. Berdasarkan hasil kuesioner terhadap mahasiswa pengguna, server docker ini telah membantu mayoritas mahasiswa dalam menghemat biaya listrik dan pembelian komputer baru, serta memudahkan para dosen untuk mengecek hasil program dari mahasiswa yang bersangkutan.

Pada tahap selanjutnya, penelitian ini dapat dikembangkan berupa transformasi server agar bisa menjadi cluster, dimana cluster tersebut dapat terhubung dengan server lain dan menstabilkan sumber daya sehingga mahasiswa dapat menggunakan sumber daya menjadi lebih maksimal.

Ucapan Terima Kasih

Kegiatan penelitian ini terwujud atas hibah Penelitian Terapan Perguruan Tinggi dari Universitas Dian Nuswantoro Semarang, Program Doktor Ilmu Komputer dan Center of Excellence Universitas Dian Nuswantoro dalam penyediaan sumber data dan library yang dibutuhkan.

Referensi

[1]. K. H. Mok, W. Xiong, and H. N. bin Aedy Rahman, "COVID-19 pandemic's disruption on university teaching and learning and competence cultivation: Student evaluation of online learning experiences in Hong Kong," International Journal of Chinese Education, vol. 10, no. 1, p. 221258682110070, Jun. 2021, doi: 10.1177/22125868211007011.

[2]. S. Barsotti, "Higher Education Was Already Ripe for Disruption. Then, COVID-19 Happened.," https://www.cmu.edu/news/stories/archives/2020/september/higher-education-covid-disruption.html, Sep. 14, 2020.

[3]. The World University Rankings, "The impact of coronavirus on higher education," https://www.timeshighereducation.com/hub/keystone-academic-solutions/p/impact-coronavirus-higher-education, Jul. 12, 2021.

[4]. P. T. Endo, M. Rodrigues, G. E. Gonçalves, J. Kelner, D. H. Sadok, and C. Curescu, "High availability in clouds: systematic review and research challenges," *Journal of Cloud Computing*, vol. 5, no. 1, p. 16, Dec. 2016, doi: 10.1186/s13677-016-0066-8.

[5]. B. Wang, Y. Song, X. Cui, and J. Cao, "Performance comparison between hypervisor- and container-based virtualizations for cloud users," in *2017 4th International Conference on Systems and Informatics (ICSAI)*, Nov. 2017, pp. 684–689. doi: 10.1109/ICSAI.2017.8248375.

[6]. R. Khalida, A. Muhajirin, and S. Setiawati, "Teknis Kerja Docker Container untuk Optimalisasi Penyebaran Aplikasi," *PIKSEL : Penelitian Ilmu Komputer Sistem Embedded and Logic*, vol. 7, no. 2, pp. 167–176, Sep. 2019, doi: 10.33558/piksel.v7i2.1819.

[7]. A. M. Potdar, N. D G, S. Kengond, and M. M. Mulla, "Performance Evaluation of Docker Container and Virtual Machine," *Procedia Computer Science*, vol. 171, pp. 1419–1428, 2020, doi: 10.1016/j.procs.2020.04.152.

[8]. Y. Li, "Towards fast prototyping of cloud-based environmental decision support systems for environmental scientists using R Shiny and Docker," *Environmental Modelling & Software*, vol. 132, p. 104797, Oct. 2020, doi: 10.1016/j.envsoft.2020.104797.

[9]. X. Wu, S. Chen, J. Huang, A. Li, R. Xiao, and X. Cui, "DDeep3M: Docker-powered deep learning for biomedical image segmentation," *Journal of Neuroscience Methods*, vol. 342, p. 108804, Aug. 2020, doi: 10.1016/j.jneumeth.2020.108804.

[10]. R. Senington, B. Pataki, and X. V. Wang, "Using docker for factory system software management: Experience report," *Procedia CIRP*, vol. 72, pp. 659–664, 2018, doi: 10.1016/j.procir.2018.03.173.

[11]. A. Zerouali, T. Mens, and C. de Roover, "On the usage of JavaScript, Python and Ruby packages in Docker Hub images," *Science of Computer Programming*, vol. 207, p. 102653, Jul. 2021, doi: 10.1016/j.scico.2021.102653.

[12]. P. Jain, Y. Munjal, J. Gera, and P. Gupta, "Performance Analysis of Various Server Hosting Techniques," *Procedia Computer Science*, vol. 173, pp. 70–77, 2020, doi: 10.1016/j.procs.2020.06.010.

[13]. B. Balatamoghna, A. Jaganath, S. Vaideeshwaran, A. Subramanian, and K. Suganthi., "Integrated balancing approach for hosting services with optimal efficiency - Self Hosting with Docker," *Materials Today: Proceedings*, Mar. 2022, doi: 10.1016/j.matpr.2022.03.078.

[14]. E. A. Santos, C. McLean, C. Solinas, and A. Hindle, "How does docker affect energy consumption? Evaluating workloads in and out of Docker containers," *Journal of Systems and Software*, vol. 146, pp. 14–25, Dec. 2018, doi: 10.1016/j.jss.2018.07.077.

[15]. R. Morabito and N. Beijar, "A Framework based on SDN and Containers for Dynamic Service Chains on IoT Gateways," in *Proceedings of the Workshop on Hot Topics in Container Networking and Networked Systems*, Aug. 2017, pp. 42–47. doi: 10.1145/3094405.3094413.