Sinaga, A., Sinaga, A. M. and Sianturi, R. A. (2023) "Desain Arsitektur Virtualisasi Pengembangan Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga pada Bank Sampah Berbasis Container Docker", Sebatik, 27(2).
Submitted: 2023-11-24 Accepted: 2023-11-29 Published: 2023-12-15

Sebatik Vol. 27 No. 2 Desember 2023
ISSN: 1410-3737(p) 2621-069X(e)

DOI:10.46984/sebatik.v27i2.2405

DESAIN ARSITEKTUR VIRTUALISASI PENGEMBANGAN SISTEM INFORMASI MANAJEMEN PENGELOLAAN SAMPAH RUMAH TANGGA PADA BANK SAMPAH BERBASIS CONTAINER DOCKER

Ardiles Sinaga[1)], Arnaldo Marulitua Sinaga[2)], dan **Riyanthi Angrainy Sianturi[3)]**

[1,2,3]Sarjana Terapan Teknologi Rekayasa Perangkat Lunak, Fakultas Vokasi, Institut Teknologi Del
[1,2,3]Sitoluama, Toba, 22381
E-mail: ardiles.sinaga@del.ac.id[1)], aldo@del.ac.id[2)], riyanthi@del.ac.id[3)]

ABSTRAK

Sampah telah menjadi salah satu masalah di negara kita Indonesia, khususnya di daerah tempat kita tinggal hari ini. Berdasarkan data Direktorat Jenderal Pengelolaan Sampah Kementerian Lingkungan Hidup dan Kehutanan tahun 2022 jumlah timbulan sampah mencapai 19,25 juta ton/tahun, jumlah pengurangan sampah hanya 26,39% atau sekitar 5,08 juta ton/tahun, jumlah sampah yang ditangani sekitar 51,47% atau sekitar 9,91 juta ton per tahun, sampah yang dapat dikelola sekitar 77.86% atau sekitar 14,99 juta ton/tahun, sedangkan jumlah sampah yang tidak terkelola sekitar 22,14% atau sekitar 4,26 juta ton/tahun. Untuk mengatasi permasalahan pengelolaan sampah ini salah satunya adalah dengan mengembangkan program bank sampah. Bank Sampah Bersinar merupakan bank sampah yang dikelola oleh PT. Solusi Rahayu Indonesia yang sudah memiliki jejaring bank sampah yang sudah tersebar di 6 kota/kabupaten di seluruh Indonesia. Bank Sampah Bersinar membutuhkan sebuah Sistem Informasi Manajemen untuk mengelola dan mengintegrasikan semua *stakeholder* yang terlibat serta data-data yang mendukung proses pengelolaan sampah rumah tangga sehingga fungsi manajemen dari Bank Sampah Bersinar dapat berjalan dengan baik. Tujuan dari penelitian ini adalah membuat desain arsitektur virtualisasi untuk pengembangan sistem informasi manajamen pengolahan sampah rumah tangga berbasis *docker container* sehingga diharapkan aplikasi web yang dibangun dapat bekerja lebih fleksible, efisien, dan memiliki ketersediaan yang tinggi (*high availability*) jika diperlukan. Hasil penelitian ini adalah berupa desain yang dapat digunakan sebagai dasar untuk mengimplementasikan arsitektur virtualisasi sistem informasi manajemen pengelolaan sampah rumah tangga berbasis *docker*, dimana desain ini bukanlah desain akhir sebelum diimplementasikan. Tetapi desain ini masih membutuhkan pengujian lebih lanjut untuk mematangkan design ini sebelum diimplementasikan.

Kata Kunci: *Desain, Arsitektur, Virtualisasi, Container, Docker, Sistem Informasi, Bank Sampah*

1. PENDAHULUAN

Sampah telah menjadi salah satu masalah di negara kita Indonesia, khususnya di daerah tempat kita tinggal hari ini. Berdasarkan data Direktorat Jenderal Pengelolaan Sampah Kementerian Lingkungan Hidup dan Kehutanan tahun 2022 jumlah timbulan sampah mencapai 19,25 juta ton/tahun, sumber sampah paling banyak berasal dari limbah rumah tangga yaitu sebesar 39,65%.

Untuk penanganan jumlah sampah sebanyak itu, pemerintah telah menuangkan kebijakan dengan diterbitkannya Undang-Undang Nomor 18 Tahun 2008 tentang Pengelolaan Sampah serta Peraturan Pemerintah Nomor 81 Tahun 2012 yang memuat perubahan paradigma dasar terkait pengelolaan sampah yaitu paradigma yang bertumpu pada kumpul-angkut-buang menjadi pengelohan sampah yang bertumpu pada pengurangan dan penanganan sampah. Paradigma baru ini harus terus dikembangkan karena paradigma baru ini menitikberatkan bahwa sampah merupakan sumber daya yang memiliki nilai ekonomis dan dapat dimanfaatkan untuk hal-hal yang berguna seperti energi, pupuk, kompos, dan juga bahan baku industri. Untuk mengelola sampah dengan baik, dapat dimulai dari hulu yaitu sejak suatu produk belum berpotensi untuk menjadi sampah. Kemudian, dilanjutkan sampai ke hilir, dimana produk yang telah digunakan sudah menjadi sampah, yang nantinya sampah tersebut dikembalikan ke lingkungan secara aman (Suryani, 2014).

Untuk mengatasi permasalahan pengelolaan sampah telah diupayakan berbagai hal, salah satu diantaranya adalah dengan mengembangkan program bank sampah. Bank sampah adalah tempat yang digunakan untuk memilah dan mengumpulkan sampah yang dapat didaur ulang dan atau diguna-ulang sehingga sampah tersebut memiliki nilai ekonomis (Istanabi, 2022) dan memberi manfaat secara ekonomi kepada masyarakat (Suhada, 2017). Bank sampah juga merupakan suatu tempat pengelolaan sampah kering yang dikumpulkan secara kolektif yang mendorong semua lapisan masyarakat untuk dapat berperan aktif di dalamnya (Lidimilah & Hermanto, 2018). Selain itu, menurut Peraturan KLHK No.14 Tahun 2021, bank sampah merupakan suatu fasilitas yang dapat digunakan untuk mengelola sampah dengan menerapkan prinsip 3R (*Reuse, Reduce, Recycle*), dimana bank sampah juga dapat berperan

sebagai sarana edukasi, perubahan perilaku dalam pengelolaan sampah, dan pelaksanaan ekonomi sirkuler yang dibentuk dan dikelola oleh masyarakat, badan usaha dan pemerintah daerah (Darma, 2023).

Bank Sampah Bersinar merupakan bank sampah yang dikelola oleh PT. Solusi Rahayu Indonesia yang sudah memiliki jejaring bank sampah yang sudah tersebar di 6 kota/kabupaten di seluruh Indonesia melalui direktori unit-unit Bank Sampah Bersinar untuk melayani kirim (*drop*) langsung sampah dan penjemputan (*pickup*) sampah pada unit-unit bank sampah tersebut. Bank Sampah Bersinar sampai saat ini melayani 80.000 nasabah yang sudah terigistrasi, 855 bank sampah unit, dan 1.500 titik edukasi yang tersebar di 6 kota/kabupaten melalui direktori unit-unit Bank Sampah Bersinar. Untuk mengelola nasabah, bank sampah unit, dan titik edukasi sebanyak itu, Bank Sampah Besinar membutuhkan sebuah Sistem Informasi Manajemen untuk mengelola dan mengintegrasikan semua *stakeholder* yang terlibat serta data-data yang mendukung proses pengelolaan sampah rumah tangga sehingga fungsi manajemen dari Bank Sampah Bersinar dapat berjalan dengan baik.

Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga yang akan dibangun ini merupakan sebuah aplikasi berbasis web. Hal ini didasari agar aplikasi dapat diakses dari berbagai platform komputer dengan menjalankannya melalui *web browser*. Disamping itu, kemudahan proses penyebaran (*deployment*) aplikasi web yang dibangun serta perangkat lunak pendukung seperti *web server*, *server* basis data, *dependency* dan *environment* lain ke *server* sangat dibutuhkan (Dwiyatno, 2020). Sebelum Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga berbasis web siap dipublikasikan, terdapat beberapa langkah yang perlu diperhatikan salah satunya adalah penyiapan *environment* untuk menjalankan aplikasi web tersebut, misalnya server (baik server secara fisik maupun *virtual*), sistem operasi, server database, serta layanan lainnya. Dalam proses mempersiapkan *environment* ini pemilihan arsitektur aplikasi merupakan salah satu diantaranya. Jika mengabaikan ini kemungkinan dapat terjadi konflik antara aplikasi yang dibangun dengan aplikasi lainnya, ini terjadi karena aplikasi-aplikasi tersebut berjalan di satu mesin. Dimana hal ini dapat menimbulkan masalah besar yang menyebabkan sistem lumpuh, lalu masalah kedua yang muncul adalah saat proses *development* adalah aplikasi yang dikembangkan lebih dari satu, hal ini kurang efisien dan sangat merepotkan karena harus menginstall semua *tools* dan *platform* yang diperlukan oleh semua aplikasi yang sedang berjalan pada komputer *developer*. Selanjutnya masalah ketiga yaitu masalah keamanan, aplikasi yang kita bangun tentunya membutuhkan keamanan *extra* untuk melindungi aplikasi dari serangan-serangan yang masuk. Masalah lainnya adalah aplikasi cukup sulit untuk diperbesar atau diperkecil sesuai dengan kebutuhan pengguna. Hal ini tidak memungkinkan kita untuk terus menerus melakukan *upgrade hardware* dari *server*, karena itu akan berdampak besar pada biaya yang ditimbulkan (Sutanto, 2021). Berdasarkan pemasalahan di atas kita membutuhkan teknologi yang tepat untuk menangani ini semua.

Untuk mengatasi permasalahan pengembangan web di atas, khususnya terkait dengan *deployment* aplikasi, kita dapat memanfaatkan teknologi virtualisasi. Virtualisasi merupakan kebalikan dari mesin fisik, dimana dapat berfungsi sebagai komputasi yang layaknya komputer yang dapat menjalankan program, dimana sebenarnya virtualisasi berjalan di atas mesin lain (Isron, 2021). Secara umum virtualisasi ada dua jenis, yaitu mesin *virtual* berbasi *hypervisor* dan virtualisasi *container*. Virtualisasi berbasis *hypervisor* membutuhkan sumber daya yang cukup besar, karena setiap *virtual machine* menjalankan *guest OS* serta kernalnya sendiri terpisah dari *host*. Sehingga ketika menjalankan aplikasi yang kemungkinan besarnya hanya 10 MB, VM kemungkinana harus menjalankan OS yang besarnya bisa mencapai 10 GB (Bik, 2017). Sedangkan virtualisasi berbasis *container* dapat menjadi sebuah titik temu untuk kinerja aplikasi yang terkait dengan aksesibilitas dan juga skalabilitasi (Raghavendra, 2018). Teknologi baru dari virtualisasi *container* yang dapat diadopsi adalah *containerization* yang merupakan teknologi yang yang mengemas aplikasi, *libary* sistem yang terorganisir, dan dependensi yang terkait untuk membangun layanan yang diinginkan dalam bentuk wadah atau *container*. Dimana aplikasi yang dibangun dan diatur dapat dijalankan dan juga digunakan sebagai sebuah wadah (Potdar, 2020).

Teknologi *container* yang popular saat ini adalah *Docker*. *Docker* merupakan salah satu proyek open-source yang memberi kesempatan kepada *developer* untuk membangun, mengemas, dan menjalakan aplikasi dalam sebuah wadah datau *container* secara terbuka (Apridayanti, 2018). *Docker* memiliki arsitektur *client-server*, dimana *client* dapat mengirimkan *request* ke *docker daemon* untuk membangun, mendistribusikan, dan menjalankan *docker container* (Khalida, 2019). *Docker* meminimalisir penyediaan *resource* yang tidak dibutuhkan dalam *deployment* aplikasi. *Deployment* pada *docker* memiliki *scalability* tinggi yang dapat diskalakan secara horizontal, *server* dapat direplikasi menjadi beberapa *node* sehingga apabila *server* yang lain mengalami *crash* masih ada *server* lain yang melayani *request*. Kemampuan lain dari *docker* yang menggunakan konsep *container* ini salah satunya adalah dapat memeriksa *container* dan versinya yaitu untuk membedakan dua atau lebih *container* sehingga aplikasi dapat diperbaiki dengan mudah jika diperlukan (Bellishree, 2020).

Berdasarkan anasilis yang sudah dijabarkan di atas, pada penelitian ini akan membahas mengenai desain arsitektur virtualisasi untuk pengembangan sistem informasi manajamen pengolahan sampah rumah tangga

berbasis *docker container* sehingga diharapkan aplikasi web yang dibangun dapat bekerja lebih fleksible, efisien, dan memiliki ketersediaan yang tinggi (*high availability*) jika diperlukan.

2. RUANG LINGKUP

Ruang lingkup pembahasan dari penelitian ini adalah membuat desain arsitektur virtualisasi berbasis *container docker* pada Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga untuk mempersiapkan *environment* sistem yang dijalankan dalam lingkungan aplikasi berbasis web yang diharapkan dapat meningkatkan performa dari server yang digunakan dan mempermudah proses *deployment* aplikasi web dan juga perangkat lunak pendukung lainnya seperti *web server*, *database*, dan juga komponen lainnya ke *server*.

Penelitian ini memiliki beberapa batasan permasalahan yaitu yang pertama penelitian ini akan membuat desain arsitektur virtualisasi aplikasi web yang dibangun menggunakan teknologi *container docker*. Kedua, desain arsitektur aplikasi yang akan dibuat dirancang pada *server virtual*. Ketiga, sistem informasi yang akan di*deploy* pada server virtual adalah aplikasi berbasis web yang dibangun menggunakan *framework* Laravel dan RESTFul API menggunakan GoLang. Dan keempat, *virtual server cloud* yang digunakan adalah Amazon EC2 instance.

Rencana hasil yang diinginkan dari penelitian ini adalah desain arsitektur virtualisasi sistem informasi manajemen pengelolaan sampah rumah tangga pada Bank Sampah Bersinar menggunakan *container docker*.

3. BAHAN DAN METODE

Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga ini merupakan suatu sistem informasi yang digunakan oleh Bank Sampah Bersinar untuk mengelola data-data nasabah, data-data transaksi penerimaan dan pengeluaran sampah, dapat pembelian dan penjualan sampah, dan data-data administratif lainnya yang dapat mempermudah bank sampah untuk mengelola semua itu. Agar sistem informasi ini dapat diakses dan digunakan kapanpun dan dimanapun, sistem informasi ini harus di *deploy* ke lingkungan yang dapat diakses melalui internet dan juga ditempatkan pada lingkungan yang mudah dikelola. Tahap awal yang perlu dilakukan adalah merancang desin arsitektur dari sistem informasi manajemen bank sampah ini.

Perancangan desain arsitektur sistem informasi manajemen pengelolan sampah rumah tangga ini menggunakan metode *System Development Life Cycle* (SDLC) yang dapat dijabarkan sebebagi berikut:

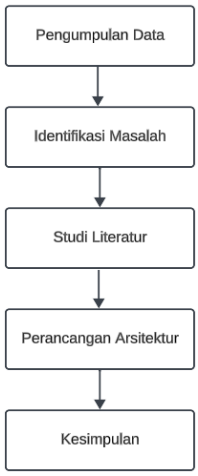
1. Pengumpulan Data
   Pada tahapan ini, teknik pengumpulan data yang dilakukan dengan observasi langsung terkait dengan sistem informasi manajemen bank sampah yang ada pada Bank Sampah Bersinar, wawancara dengan pihak-pihak terkait yang saling berkaitan dengan sistem ini seperti nasabah baik itu individu maupun bank sampah unit (komunitas), petugas lapangan maupun petugas yang berada di kantor, pengelola sistem informasi manajemen, dan *offtaker* yang membeli sampah-sampah yang dikumpulkan oleh Bank Sampah Bersinar, dan titik edukasi yang tersebar di beberapa titik.
2. Identifikasi masalah
   Pada tahapan ini, ada beberapa tindakan yang dilakukan untuk mengidentifikasi permasalahan yaitu:
   1) Mempelajari aliran kerja sistem informasi mulai nasabah (individu maupun komunitas) kemudian dikumpulkan petugas atau diantar ke Bank Sampah oleh nasabah, masuk ke gudang untuk diproses dan disimpan, dan kemudian dijual ke *offtaker* (lapak atau pabrik).
   2) Mempelajari ketentuan-ketentuan dan aturan-aturan yang berlaku terkait dengan pengelolaan sampah pada Bank Sampah Bersinar.
   3) Mempelajari aliran data mulai dari pengumpulan sampah, pengelolaan sampah, penjualan sampah, dan data-data untuk menyusun laporan keuangan.
   4) Menganalisa sistem pengelolaan sampah pada Bank Sampah Bersinar dengan membuat rancangan sistem berdasarkan data yang sudah dikumpulkan sebelumnya.
3. Studi Literatur
   Pada tahap ini hal yang dilakukan adalah mempelajari mengenai teknologi virtualisasi berbasis *container docker* secara sistematis dan cermat mengenai fakta-fakta yang terkait dengan teknologi tersebut melalui sumber-sumber yang berkaitan dengan penelitian ini baik itu dari buku-buku referensi, *text book*, ataupun penelitian-penelitian terdahulu yang dapat digunakan untuk mendukung penelitian ini.
4. Perancangan Arsitektur Sistem Informasi Manajemen
   Dalam perancangan arsitektur ini digunakan metode System Development Life Cycle (SDLC). Namun tidak semua tahapan pada metode ini yang digunakan, karena pada penelitian ini hanya sampai membuat desain saja.
   1) Perencanaan Arsitektur Sistem
      Tujuan pada tahap perencanaan ini untuk menentukan dan mendefenisikan arsitektur seperti apa yang akan digunakan pada sistem informasi manajemen pengelolaan sampah rumah tangga ini. Pada tahap ini juga dikumpulkan beberapa perangangkat lunak dan server yang akan digunakan untuk membuat arsitektur ini.
   2) Analisis Arsitektur Sistem
      Pada tahap ini dilakukan analisis arsitektur sistem yang akan digunakan pada sistem informasi ini. Dari hasil analisis ini apa saja

yang dibutuhkan untuk menganalisa dan membuat desain arsitektur virtualisasi pada sistem informasi manajemen tersebut menggunakan teknologi *container docker*.

3) Rancangan Arsitektur
Tahapan perancangan pada penelitian ini dilakuakn untuk membuat desain arsitektur virtualisasi berbasis *container docker*. Hasil dari rancangan ini merupakan sebuah desain model arsitektur virtualisasi berbasis docker yang direncanakan akan menjadi desain usulan arsitektur sistem informasi manajemen pengeloaan sampah rumah tangga menggunakan VPS (*Virtual Private Server*) sebagai servernya dan di dalam server tersebut akan diinstall aplikasi Docker untuk membuat container yang berisi aplikasi dari sistem informasi tersebut, *database*, *web server*, dan juga perangkat-perangkat lunak lainnya yang mendukung sistem informasi yang dikembangkan ini.

5. Kesimpulan
Tahap terakhir dari penelitian ini adalah untuk menyimpulkan hasil yang dicapai dari tahap-tahap yang sudah dilakukan sebelumnya.

Agar lebih jelas dan lebih mudah dalam membaca langkah-langkah pengembangan arsitektur virtualisasi berbasis *container docker* di atas, dapat dilihat dalam bentuk *flowchart* seperti Gambar 1.



Gambar 1. Tahapan Penegmbangan Arsitektur Virtualisasi Berbasis *Container Docker*

3.1 Analisis Kebutuhan Data

Tahapan awal yang dilakuan dalam membuat desain arsitektur virtualisasi ini adalah analisis kebutuhan data. Data-data yang dikumpulkan dalam membangun desain ini dalah:

1. Data-data terkait dengan *images docker* yang akan diinstal ke dalam arsitektur yang akan dibangun pada penelitian ini. *Image-image docker* ini akan diinstal ke masing-masing *container*.
2. Jumlah dari aplikasi web yang akan dijalankan pada arsitektur virtual ini, dirancanakan akan menjalankan 2 aplikasi web yang akan berperan sebagai aplikasi *front-end* dan aplikasi *back-end*.
3. Data berupa *network* yang akan mengisolasi masing-masing aplikasi, sehingga diharapkan dapat membatasi hak akses dari pengguna untuk mengakses aplikasi tersebut.

3.2 Virtualisasi Server

Virtualisasi jika didefenisikan secara sederhana memiliki lawan kata dari mesin fisik atau *physical machine*. *Physical machine* merupakan perwujudan dari server yang terdiri dari komponen-komponen seperti *disk*, *power suplay*, *mainboard*, dan sebagainya. Dalam defenisi lainnya, virtualisasi juga diartikan sebagai aplikasi seolah-olah hanya dijalankan dalam satu mesin saja, tetapi yang terjadi sebenarnya bahwa virtualisasi itu berjalan juga di mesin yang lainnya secara bersama-sama dengan aplikasi yang lainnya (Khalida, 2019).

Virtualisasi juga merupakan sebuah teknik yang dapat digunakan untuk menyembunyikan karakter fisik dari sumber daya komputer dari pengguna, aplikasi, serta bagaimana sistem lain berinteraksi dengan sumber daya dari kompoter tersebut. Termasuk di dalamnya untuk menjadikan komputer menjadi sebuah sumber daya tunggal misalnya seperti sebuah *operating system*, *server*, sebuah aplikasi, atau juga peralatan penyimpanan lainnya agar terlihat seperti berfungsi sebagai sumber daya logikal atau juga untuk membuat berbagai sumber daya fisik (seperti *server* atau beberapa *tool* penyimpanan) menjadi kelihatan sebagai satu sumber daya logikal (Apridayanti, 2018).

Virtualisasi/*virtualization* adalah teknik untuk membuat sesuatu menjadi bentuk yang tidak seperti kenyataan atau membuat sesuatu ke dalam bentuk virtualisasi. Virtualisasi dapat digunakan untuk membuat simulasi perangkat fisik dari sebuah komputer menjadi seolah-olah perangkat komputer tersebut disembunyikan (menjadi tidak ada) atau membuat perangkat yang seharusnya tidak ada menjadi ada.

3.3 Container

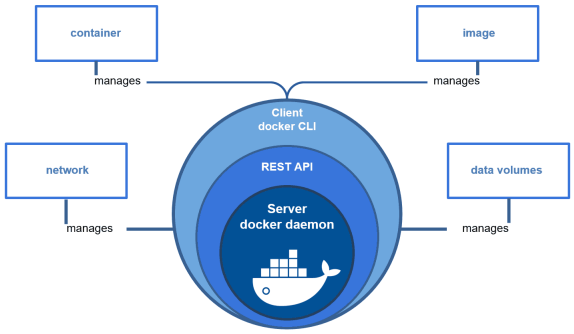
Container merupakan sistem operasi yang ringan (*lightweight*) yang secara langsung dapat bekerja di dalam *host* sistem operasi. *Container* dapat menjalankan secara langsung segala proses-proses intruksi ke *core* CPU. Container juga memiliki kemampuan untuk menghemat penggunaan sumber daya (*resource*) tanpu terjadi *overhead* dari *virtualization* dan juga dapat menjamin kinerja dari aplikasi yang terisolasi (Fihri, 2019). Jika dibandingkan antara *container* dengan *virtualization*, ukuran *container* jauh lebih kecil deibandingkan *virtualization* karena biasanya ukuran *container* hanya berskala *megabyte*, dimana *container* itu

biasanya berisi paket aplikasi yang dibutuhkan tanpa ada sistem operasi yang berjalan di dalamnya. Performansi dari aplikasi dapat berjalan lebih baik tanpa adanya *hypervisor*, terutama jika proses interaksi dengan I/O *devices* dibutuhkan oleh aplikasi tersebut (Zhang, 2018).

3.4 Docker

Docker merupakan teknologi yang tersedia secara terbuka untuk *developer* atau juga *sysadmin* untuk dapat membangun, mengemas, dan menjalankan aplikasi di manapun sebagai sebuah wadah ringan yang biasanya disebut dengan *container*. Ini berbeda dengan virtualisasi yang menjalankan aplikasi di atas *hypervisor* dan guest OS. Sedangkan pada *docker*, aplikasi dapat berjalan secara langsung tanpa kedua hal yang dilakukan oleh hypervisor yang membuat *Docker* menjadi lebih hemat dalam menggunakan sumber daya server (Sutanto, 2021). Desain arsitektur yang dimiliki oleh *Docker* memberi kemudahan untuk mendistribusikan dan juga mengembangkan aplikasi dengan lebih cepat karena *Docker* mempunyai sifat *Lightweight Containerization* yang dilengkapi dengan bermacam-macam komponen serta fitur sehingga mampu memberi kemudahan bagi para developer untuk mengembangkan dan juga mamantau kinerja aplikasi yang diciptakan (Fihri, Negara & Sanjoyo, 2019).

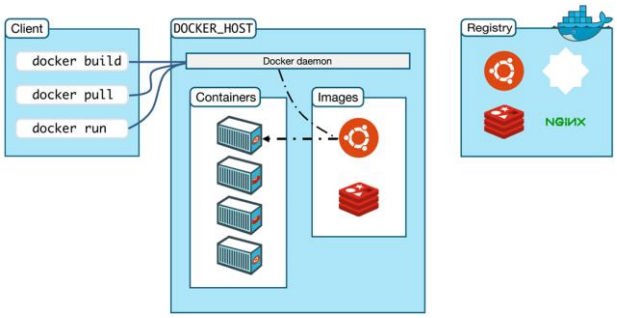
Untuk membuat objek, *Docker* dapat dikonfigurasikan. Objek-objek yang terdapat pada *Docker* yaitu *container*, *service*, dan *images*. Objek-objek tersebutlah yang menjadi orkestrasi dalam proses *containerization*. Dibutuhkan file instruksi yang disebut dengan *dockerfile* untuk menerapkan aplikasi ke dalam platform *docker* tanpa harus mengubah apapun pada infrastruktur dari aplikasi yang tidak dibutuhkan (Li, 2020). Pada host yang sama, *client docker* dan *docker daemon* dapat berjalan dan berkomunikasi melalui soket maupun jaringan dengan menggunakan permintaan (request) API). Kita dapat melihat komponen-komponen yang dimiliki oleh *docker engine* pada Gambar 2 di bawah ini:



Gambar 2. Komponen-komponen Docker Engine (Dwiyatno, dkk., 2020)

Pada saat men*deploy* aplikasi *docker* dapat meminimalkan penyediaan *resource* yang tidak diperlukan. Disamping itu, *docker* memiliki portabilitas yang tinggi terkait dengan *uptime* dan juga *deployment* aplikasi. Waktu yang dibutuhkan untuk menyediakan layanan (*uptime*) oleh layanan *container* hanyalah beberapa detik saja, *docker* juga dapat menyimpan image (bahan *deployment*) ke dalam *docker hub* (repositori *online*) agar pada suatu waktu dapat diunduh jika diperlukan, lalu diimplementasikan ke dalam *container* untuk mempermudah pada saat *production* ke *server*. Skalabilitas yang tinggi yang dimiliki oleh *docker* membuat proses *deployment* aplikasi dapat diskalakan menjadi beberapa server (secara horizontal), *docker* juga dapat mereplikasi server menjadi beberapa *node* yang berfungsi untuk melayani *request* yang dilakukan oleh server lain jika server lainnya mengalami *crash* (Adiputra, 2015).

Arsitektur *docker* terdiri dari banyak komponen-komponen diantranya adalah *docker images*, *docker daemon*, *docker container*, *docker registry*, dan *docker client*. Teknologi *client-server* yang digunakan oleh *docker* untuk menghubungkan *docker daemon* dan *docker client* adalah teknologi *client-server*. Untuk lebih jelasnya, arsitektur *docker* dapat dilihat pada Gambar 3 di bawah ini:



Gambar 3. Arsitektur Docker (Dwiyatno, dkk., 2020)

Setiap komponen-komponen pada *docker* menjalankan proses dan tugas masing-masing. Komponen-komponen tersebut saling bergantung satu dengan yang lainnya, sehingga ketika salah dari komponen-komponen itu tidak tersedia, itu dapat memberi pengaruh terhadap kinerja dari *docker* itu sendiri.

Di bawah adalah komponen-komponen yang dimiliki oleh *docker* (Dwiyatno, dkk., 2020):

1. *Docker images* merupakan salah satu komponen yang dimiliki oleh *docker* yang berupa *template* dasar yang sifatnya *read only*. Pada dasarnya template ini adalah sebua sistem operasi atau sistem operasi yang sudah di-*install* bermacam-macam aplikasi. *Docker images* memiliki fungsi untuk menciptakan *docker container*, hanya

dengan satu *docker images* yang dibuat, mampu membuat banyak *docker container*.

2. *Docker daemon* merupakan sebuah layanan (*service*) yang dioperasikan (berjalan) di dalam host atau sistem operasi. Komponen ini berfungsi untuk menjalankan dan mendistribusikan *container docker*. Secara tidak langsung pengguna tidak dapat mengunakan *docker daemon*, namun pengguna dapat menggunakan *docker client* untuk menjalankan *docker client* sebagai perantara atau CLI.
3. *Docker client* merupakan sekumpulan perintah *command line* yang digunakan untuk mengoperasikan *docker container*, contohnya untuk membuat *container*, menghapus (*destroy*) *container*, *start/stop container*, dan melakukan hal lain yang terpat pada *docker container*.
4. *Docker registry* merupakan komponen *docker* yang berfungsi sebagai *repository* distribusi kumpulan dari *docker images* yang memiliki sifat *private* atapun *public* yang mampu diakses melalui *docker hub*.
5. *Docker container* merupakan sebuah *images* yang bersifat *read-write* dan dapat dikemas, dimana *container* ini berjalan di atas *images*. *Docker container* memiliki definisi lain sebagai sebuah folder yang dibuat menggunakan *docker container*. Setiap *container* baru yang dibuat dan kemudian disimpan akan membentuk layer baru tept berada di atas *docker images* atau *base images* di atasnya.

3.5 Bank Sampah

Bank Sampah merupakan suatu kegiatan yang sifatnya *social engineering* untuk membari edukasi kepada masyarakat terkait pemilahan sampah dalam menumbuhkan kesadaran masyarakat tentang pentingnya pengeloalaan sampah secara bijak yang nantinya akan berdampak untuk mengurangi sampah yang akan diangkut ke Tempat Pembuangan Akhir (TPA) (Yunita et al., 2021).

Cara kerja dari dari bank sampah itu sendiri hampir mirip dengan bank konvensional pada umumnya, dimana bank sampah itu sendiri terdiri dari nasabah, yang di dalmnua juga ada pencatatan pembukuan dan juga yang paling penting adalah manajemen pengelolaannya. Perbedaan antara bank sampah dan bank pada umumnya terletak pada apa yang disetorkan. Kalau pada bank pada umumnya yang disetorkan adalah uang, sedangakan pada bank sampah yang disetorkan adalah sampah yang memiliki nilai ekonomis (Arifin, 2020).

3.6 Sistem Informasi

Sistem informasi adalah suatu kombinasi yang teratur yang melibatkan jaringan komunikasi, pengguna (orang-orang), perangkat lunak, dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi. Fungsi dari sistem informasi ini salah satunya adalah meningkatkan aksesibilitas data yang sudah disajikan sebelumnya dengan tepat waktu dan akurat untuk para *user* (Anggraen, 2017). Kegiatan-kegiatan yang ada pada organisasi dikelola oleh sistem informasi secara terkomputerisai untuk mengahasilkan informasi-informasi yang diperlukan oleh *stakeholder* yang berada di dalam ataupun di luar organisasi tersebut. Untuk mengoptimalkasn efisiensi pengelolaan sumber daya publik maka dibutuhkan sistem informasi berbasis web (Melchor, 2016).

4. PEMBAHASAN

Pada bagian pembahassan ini akan dijabarkan terkait dengan tahapan yang dilakukan pada penelitian ini, dinataranya adalah:

4.1 Analisa Sistem

Pada sub bab ini akan dibahas mengenai analisa sistem dari sistem informasi manajaemen yang akan dibuatkan desain arsitekturnya dengan cara menguraikan sistem informasi yang dibangun secara utuh ke dalam bagian komponennya masing-masing dengan maksud untuk melakukan identifikasi dan evaluasi terhadap kelemahan dan kelebihan yang ditemukan pada sistem tersebut.

Semua aplikasi berbasis web berserta environment masing-masing yang dibutuhkan akan di *deploy* ke dalam *container*. Masing-masing *container* akan mempunyai alamat IP *Private* yang hanya dapat diakses oleh *Host*. Jika ingin dapat diakses oleh *Host* dari luar *container*, maka harus dibuatkan domain. Kemudian *Nginx* yang berfungsi sebagai *reverse proxy* digunakan untuk mengarahkan *domain* ke *container* tujuan.

4.2 Kebutuhan Alat

Spesifikasi perangkat yang diperlukan untuk mendukung penelitian dalam melakukan perancangan arsitektur virtualisasi sistem informasi manajemen pengelolaan sampah rumah tangga adalah menggunakan *instance Amazon EC2* dengan spesifikasi sebagai berikut:

1. Processor Intel C6g.large (2vCPU)
2. RAM 4 GiB
3. 40 GB EBS Storage
4. OS Linux
5. 20 GB data out

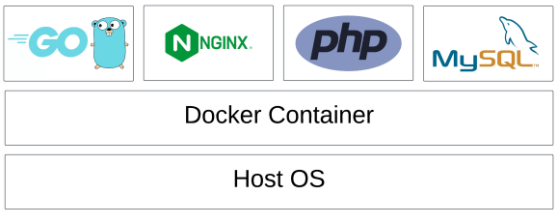
Sedangkan perangkat lunak yang dibutuhkan untuk mendukung penelitian dalam melakukan perancangan arsitektur virtualisasi sistem informasi manajemen pengelolaan sampah rumah tangga adalah:

1. *Docker Package* yang digunakan untuk menginstall, membangun, dan mengelola aplikasi dalam *container*.
2. WSL2 (*Windows Subsystem for Linux*) digunakan sebagai media untuk berkomunikasi antara *backend* linux dengan *host* OS pada Windows.

3. *Visual Studio Code* sebagai *code editor* yang digunakan untuk menulis kode program.

4.3 Arsitektur Virtual Private Server Pengembangan Sistem Informasi Manajemen

Arsitektur pengembagangan sistem informasi manajemen pengelolaan sampah yang akan dikembangkan pada penelitian ini adalah arsitektur virtualisasi berbasis *docker*. Virtualisasi VPS (*Virtual Private Server*) memberikan dampak pada proses pengembangan sistem informasi manajemen pengelolaan bank sampah yang modern, terisolasi, dan efisien. Gambar 4 merupakan gambar arsitektur VPS pengembangan sistem informasi manajemen pengelolaan bank sampah:

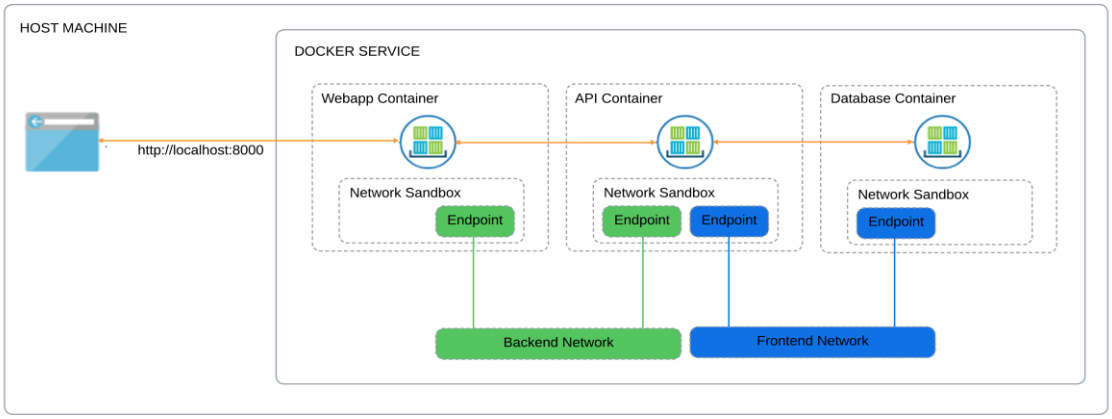


Gambar 4. Arsitektur Virtual Private Server Pengembangan Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga

Virtualisi *Virtual Private Server* pengembangan sistem informasi manajemen yang akan digunakan sebagai wadah/tempat pembuatan Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga berbasis *Docker Container*. *Image* yang dibangun dari *docker* terdiri dari php dengan *Nginx* sebagai *web server* untuk menjalankan aplikasi berbasis web yang menggunakan bahasa pemrograman php dengan Laravel sebagai *framework*nya. Selain php, *image* selanjutnya yang dibangun dari *docker* memuat golang untuk menjalan RESTful API menggunakan bahasa pemrograman Go. Kemudian selanjutnya adalah mysql yang digunakan sebagai sistem manajemen *database relational* (RDBMS). Konfigurasi *image* yang dibangun pada *docker* memuat masing-masing *environment* yang telah dirancang, kemudian setelah itu akan dilakukan proses *build image* melalui *dockerfile* tersebut yang dijalankan pada *docker*. Masing-masing *image* itu didefenisikan dalam *docker compose* untuk mengintegrasikan multi *container* pada *docker*.

4.4 Desain Arsitektur Virtualisasi Berbasis Docker

Sistem yang akan dibangun ini merupakan Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga pada bank sampah yang di bangun menggunakan docker container. Sistem ini terdidi dari beberapa aplikasi yang saling terkoneksi satu sama lainnya. Seluruh proses sistem yang bekerja berada pada *Virtual Private Server* yang dibangun dengan mengimplementasikan teknologi virtualisasi server pada *containerization* untuk membantu proses isolasi terhadap sistem aplikasi yang bersifat kompleks dan juga memungkinkan untuk *scaling system* sesuai dengan perancangan dan kebutuhan server sesuai dengan kapasitas dan spesikasi yang dimiliki. Desain sistem informasi manajemen pengelolaan sampah rumah tangga yang dibangun terdiri dari aplikasi web yang dibangun menggunakan Laravel sebagi *front-end system*, MySQL sebagai *back-end system*, dan RESTful API yang dibangun menggunakan GoLang dimana menjadi penghubung antara *front-end* dan *back-end system* yang bekerja saling terintegrasi untuk menciptakan layanan dengan performa terbaik. Berikut ini adalah desain arsitektur virtualisasi berbasis *docker* dari sistem informasi manajemen pengelolaan sampah rumah tangga:



Gambar 5. Arsitektur Virtualisasi Berbasis Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga

Gambar 5 merupakan tumpukan aplikasi *3-tier* klasik dimana aplikasi web dan RESTful API berada dalam satu jaringan dan kemudian RESTful API yang sama dihubungkan ke jarangan lain yang mempunyai server database.

Pengembangan aplikasi ini tidak merepotkan untuk penerapannya dengan jaringan fisik, *firewal*, dan lain sebagainya. Memisahkan infrastruktur di aplikasi-aplikasi secara signifikan dapat meningkatkan portabilitas dari aplikasi yang dapat didistribusikan. Dengan kata lain, pengembang dapat memiliki lebih banyak kebebabasan dalam mendefenisikan topologi dari aplikasi secara tepat.

1. Database Container

Database container merupakan bagian yang memiliki *variable environment* yang memberitahukan *port container* untuk menginisialisasi *server database* dengan nama database, nama pengguna, dan kata sandi yang sudah diatur sebelumnya.

2. API Container

Layanan API ini memiliki saluran yang memberitahukan *host container* untuk membangun *image* api yang berada di *backend* directori. Konfigurasi *networks* mempunyai 2 buah jaringan *network-backend* dan *network-frontend*. Pada gambar kita dapat melihat bahwa *container* API dan database berbagi jaringan yang sama yaitu *network-banckend.* Karena *container* API dan database berada pada jaringan yang sama, maka kedua *container* tersebut dapat saling berkomunikasi.

3. Webapp Container

Pada gambar kita dapat melihat konfigurasi baru dari ports yang memiliki format *n:m*. Fungsinya hanya memetakan *port container m* ke *port host n*, sehingga dapat mengekpos aplikasi ke host. Pada gambar *node* aplikasi berjalan pada port 8000 di dalam *container*.

Perhatikan juga pada gambar yang menunjukkan bahwa konfigurasi dari *networks* hanya memiliki *network frontend*. Perlu diperhatikan juga bahwa *network frontend* juga digunakan bersama oleh *container Webapp* dan *API*, ini berarti *Webapp* dan *API* berjalan pada jaringan yang sama dan *Webapp* dapat mengakses API tetapi tidak dapat mengakses database.

4.5 Kelemahan dan Keterbatasan Arsitektur Virtualisasi Berbasis Docker Pada Sistem Yang Diusulkan

Selain keunggulan yang dimiliki oleh *container docker* ini, ada beberapa kekurangan yang dimiliki oleh arsitektur virtualisasi berbasis *container docker* ini diantaranya:

1. Mempertahankan data pada *container docker* itu rumit. Hal ini dapat berikat fatal jika *container* dimatikan. Resiko kehilangan data sangat mungkin terjadi, data yang hilang seperti data nasabah, data transaksi, dan data lainnya tidak dapat dipulihkan jika terjadi demikian. Untuk mengatasi permasalahan ini, semua data dapat disimpan di tempat lain dan dapat diambil kembali. Cara menyimpannya dapat dilakukan dengan menyimpan data ke dalam *Docker Data Volumes* dan juga disarankan untuk melakukan *backup* data dari basis data secara berkala. Tetapi, hal ini juga cukup menyulitkan untuk dilakukan karena memiliki banyak tantangan saat menyimpan data di dalamnya.
2. Ketergantungan Platform. *Docker* biasanya tidak platform independen, tapi *docker* dapat dijadikan menjadi *Platform* Independen, namun untuk melakukannya diperlukan lapisan tambahan antara *Host* dan *Docker*. *Docker* dirancang hanya untuk digunakan pada Linux seperti *halnya virtual private server* yang digunakan pada sistem inforamsi inimenggunakan Linux Ubuntu Server. Namun, saat ini *docker* juga dapat digunakan pada sistem operasi lainnya seperti *Windows*, Mac OX, dan sebagainya dengan menerapkan lapisan tambahan.
3. Masalah Kompabilitas. Jika *docker* diimplementasikan ke mesin lokal, *Docker* kompatibel dengan mesin lokal yang memiliki arsitektur 64-bit. *Docker* tidak mendukung mesin lama yang kurang dari 32-bit.
4. *Docker* menghabiskan cukup besar sumber daya yang tersedia dan menggunakannya secara efisien. Disamping tu, *Docker* juga menggunakan jaringan *overlay* untuk mengurangi kinerja *container* yang tidak sesuai.

5. KESIMPULAN

Desain arsitektur virtualisasi berbasis *docker* untuk pengembangan sistem informasi manajemen pengelolaan sampah rumah tangga pada bank sampah sudah selesai dirancang. Berdasarkan hasil yang diperoleh, maka *design* yang dihasilkan ini bukanlah desain akhir yang nantinya dapat diterapkan langsung sebagai desain virtualisasi untuk mempersiapkan *environment* sistem yang dijalankan dalam lingkungan aplikasi berbasis web yang nantinya diharapkan dapat mempermudah proses *deployment* aplikasi web dan juga perangkat pendukung seperti *database*, *web server*, dan komponen-komponen lainnya. Dalam desain tersebut pemisahan aplikasi-aplikasi seperti database, web server, aplikasi web, RESTful API, dan komponen lainnya dipisah dalam container yang berbeda yang memungkinkan penggunaan sumber daya pada masing-masing container dapat lebih efisien. Disamping itu, aplikasi-aplikasi juga dipisahkan dalam *network* yang berbeda, dimana hal ini dapat mengisolasi sistem aplikasi yang bersifat kompleks dan juga memungkinkan untuk *scaling system* sesuai dengan spesifikasi dan kapasitas server pada saat dibutuhkan. Serta diharapkan juga dengan adanya desain ini dapat meningkatkan kemampuan aplikasi web untuk mampu bekerja lebih fleksibel, efisien, dan memiliki ketersedian yang tinggi (*high availability*) jika diperlukan. Selanjutnya, desain arsitektur virtualisasi ini perlu diuji, agar diperoleh desain akhir yang dapat dimplementasikan untuk sistem informasi manajemen ini.

6. SARAN

Arsitektur virtualisasi yang dibangun ini bukanlah desain akhir. Desain ini masih perlu dilakukan pengujian terkait

dengan performa yang dihasilkan melalui rancangan yang dihasilkan pada penelitian ini. Pengujian performa di beberapa spesifikasi server berbeda dapat dilakukan untuk memperoleh hasil yang diharapkan. Spesifikasi server yang dimaksud disini adalah server yang berjalan di platform sistem operasi Windows, spesifikasi server yang lebih tinggi dari server *instance Amazon EC2* yang digunakan pada penelitian ini, dan juga server yang menggunakan mesin lokal yang memiliki arsitektur 64-bit. Sehingga diperoleh hasil pengujian yang dapat diterima dan diklaim bahwa desain yang dibuat dapat berjalan di spesifikasi server tertentu. Dengan demikian, desain akhir yang dipilih menjadi desain yang benar untuk mengimplementasikan Sistem Informasi Manajemen Pengelolaan Sampah Rumah Tangga pada Bank Sampah Bersinar. Perlu juga dipertimbangkan pendekatan atau teknologi alternatif terkait dengan jenis virtualisasi yang lainnya untuk mempersiapkan *environment* aplikasi sistem informasi pengelolaan sampah rumah tangga ini. Apakah teknologi virtualisasi alternatif tersebut memiliki performa yang lebih baik dibandingkan teknologi virtualisasi berbasis *docker container* ini.

7. DAFTAR PUSTAKA

Adiputra, F. (2015). Container dan Docker: Teknik Virtualisasi Dalam Pengelolaan Banyak Aplikasi Web. Simantec, 4(3), 167-176. DOI: https://doi.org/10.21107/simantec.v4i3.1384

Potdar, A. M., Narayan, D. G., Kengond, S., & Mulla, M. M. (2020). Performance Evaluation of Docker Container and Virtual Machine. In Procedia Computer Science (Vol. 171, pp. 1419–1428). Elsevier B.V. https://doi.org/10.1016/j.procs.2020.04.152.

Anggraeni, E. Y., Irviani, R. (2017). Pengantar Sistem Informasi. Yogyakarta: Andi.

Apridayanti, S., Isnawaty & Adi S., R. (2018). Desain dan Implementasi Virtualisasi Berbasis Docker Untuk Deployment Aplikasi Web. semanTIK, 4,(2), 37-46.

Bellishree P., & Deepamala. N. (2020). A Survey on Docker Container and its Use Cases. IRJET, 07(07), 2716-2720.

Bik, M., F., R. (2017). Implementasi Docker Untuk Pengelolaan Banyak Aplikasi Web (Studi Kasus : Jurusan Teknik Informatika UNESA). Jurnal Manajemen Informatika. 7(2), 46-50.

Brilian, R., P., Rohman, A. (2022). Sistem Informasi Manajemen Tabungan Pada Bank Sampah Raflesia Menggunakan Metode Waterfall. JBMI, 19(3), 192-204.

Dharma, A. B., Susanti, D., & Marlinda, P. (2023). Implementasi Kebijakan Sistem Informasi Manajemen Bank Sampah Di Kota Dumai. Sebatik, 27(1), 145-154. DOI:10.46984/sebatik.v27i1.2098

Dwiyatno, S., Rakhmat, E., & Gustiawan, O. (2020). Implementasi Virtualisasi Server Berbasis Docker Container. PROSISKO, 7(2), 165-175.

Fihri, M., Negara, R., M., & Sanjoyo, D. D. (2019). Implementasi & Analisis Performansi Layanan Web Pada Platform Berbasis Docker. e-Proceeding of Engineering, 6(2), 3996-4001.

Isron, M., H., Putra, R., E. (2021). Implementasi Virtual Server Berbasis Container Pada Sistem Informasi Geografis Cagar Budaya Mojokerto. JINACS, 3(2), 143-154.

Istanabi, T., Miladan, N., Suminar, L., Kusumastuti, Aliyah, I., Soedwiwahjono, Utomo, R. P., Werdiningtyas, R., & Yudana, G. (2022). Pengelolaan Bank Sampah sebagai implementasi Ekonomi Kreatif di Bank Sampah Guyub Rukun Dusun Madugondo, Kecamatan Piyungan, Bantul. PengabdianMu: Jurnal Ilmiah Pengabdian kepada Masyarakat, 7(3), 407-413. https://doi.org/10.33084/pengabdianmu.v7i3.2765

Khalida, R., Muhajirin, A., & Setiawati, S. (2019). "Teknis Kerja Docker Container untuk Optimalisasi Penyebaran Aplikasi". PIKSEL: Penelitian Ilmu Komputer Sistem Embedded and Logic, Vol. 7 (2), 167-176

Li, Y. (2020). Towards fast prototyping of cloud-based environmental decision support systems for environmental scientists using R Shiny and Docker. Environmental Modelling & Software, 132, 1-8. https://doi.org/10.1016/j.envsoft.2020.104797.

Lidimilah, L. F., Hermanto, H. (2018). Sistem Informasi Bank Sampah Sukorejo Berbasis Client Server. Jurnal Ilmiah Informatika, 3(1), 193-198. https://doi.org/10.35316/jimi.v3i1.474

Melchor, E. M., Carrillo, D. B. (2016). Web-Based System to Improve Resource Efficiency in University Departments. Journal of Cases on Information Technology, 18(1), 1–16.

Raghavendra M., S., Chawla, P. (2018). A Review on Container-Based Lightweight Virtualization for Fog Computing. 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 378-384. Doi: 10.1109/ICRITO.2018.8748346.

Suhada, B., Setyawan, D. (2017). Pengembangan Bank Sampah Syariah Ikhtiar Pemberdayaan Memajukan Ekonomi Kreatif (Studi Bank Sampah Cangkir Hijau). Akademika : Jurnal Pemikiran Islam. 22(2):245-265. DOI: https://doi.org/10.21107/dinar.v4i2.5072.

Suryani, A., S. (2014). Peran Bank Sampah Dalam Efektivitas Pengelolaan Sampah (Studi Kasus Bank Sampah Malang). Aspirasi: Jurnal Masalah-Masalah Sosial. 5(1):71-84. https://doi.org/10.46807/aspirasi.v5i1.447

Sutanto, S., Gunawan, W., & Faeshal, F. (2021). ARSITEKTUR CONTAINER DOCKER PADA APLIKASI EXPERT ASSIST DENGAN TEKNOLOGI NODE.JS, EXPRESS FRAMEWORK & CLOUD DATABASE NoSQL MONGODB ATLAS. Jurnal Sistem Informasi Dan Informatika (Simika), 4(1), 73–89. https://doi.org/10.47080/simika.v4i1.1189.

Yunita, Y., Adrianshyah, M. & Amalia, H., 2021. Sistem Informasi Bank Sampah Dengan Model Prototype. INTI Nusa Mandiri, 16(1), 15-24. DOI:10.33480/inti.v16i1.2269

Zhang, Qi & Liu, Ling & Pu, Calton & Dou, Qiwei & Wu, Liren & Zhou, Wei. (2018). A Comparative Study of Containers and Virtual Machines in Big Data Environment. 2018 IEEE 11th International Conference on Cloud Computing (CLOUD). IEEE. DOI: 10.1109/CLOUD.2018.00030