ICIC Express Letters
Part B: Applications
Volume **11**, Number **11**, November **2020**

 ISSN 2185-2766
pp. **1011–1018**

SLEEP QUALITY MONITORING SYSTEM BASED ON CONTAINER ORCHESTRATION

Nico Surantha, Vincent Valentine Jansen, Wincent and Sani Muhamad Isa

Computer Science Department, BINUS Graduate Program – Master of Computer Science
Bina Nusantara University
JL. K. H. Syahdan No. 9, Kemanggisan, Palmerah, Jakarta 11480, Indonesia
{ nico.surantha; vincent.jansen; wincent; sani.m.isa }@binus.ac.id

Received March 2020; accepted June 2020

Abstract. *Sleep is one of the most important factors of human and sleep disorders can cause many physical and mental health problems. Smart sleep monitoring systems have become a solution to tackle this problem, where the system needs to be accessible at any time and condition by multiple users bringing the needs of the high availability and throughput of the system. Containerization has become a trending topic in software development where the container concept has solved many problems of virtual machines such as cost reduction, fewer CPU needed, and CPU utilization has been improved. Many containers have been around since then, and the most famous widely used is Docker. With its extra lightweight, fast, and highly portable characteristics, docker can be run anywhere on all platforms like Linux and Windows distribution. There are so many container orchestrations that can be used to manage docker containers, such as Swarm, Kubernetes, and Nomad. The purpose of this research is to propose Kubernetes as the container orchestration for the implementation in the real-life scenario. The result shows us that Kubernetes gives a good result in availability, flexibility, and scalability compared to the other container orchestration in developing and delivering products.*
Keywords: Internet of Things, Sleep monitoring, Container, Container orchestration, Kubernetes

1. **Introduction.** Sleep is the best activity to replenish energy and physical condition. The ideal amount of sleep that is needed by each person can vary due to genetic factors, age, and other reasons [1]. When a person does not get enough sleep cycles, the lack of sleep can cause some symptoms such as sleep disorder and deficiency leading to physical and mental health problems, lack of productivity and focus, diseases, and a higher risk of death [2]. The sleep disorder can be prevented with achieving the right diagnosis and treatment based on the patient's medical record, and sleep stages are one of the crucial factors that counts [3]. The sleep stages of humans consist of awake, Non-Rapid Eye Movement (NREM) that consists of shallow sleep divided into stage 1 and stage 2, and Rapid Eye Movement (REM) that consists of deep sleep which is divided into stage 3 and stage 4 [4].

There are different ways to get sleep measurements for checking sleep quality based on the sleep stage, such as Ballistocardiography (BCG), Phonocardiography (PCG), and Electrocardiography (ECG). BCG is a measure of repetitive human body motion generated by the ejection of the blood into the great vessels at each cardiac cycle [5]. PCG is a heart sounds identification technique based on a detector and a computerized analyzer that capture heart sounds by using a stethoscope and microphone. These sounds emitted by the blood flow turbulence are detected in cardiac auscultation [6]. The last is ECG, which is acquired from the electrical activity produced by the heart activity over time

DOI: 10.24507/icicelb.11.11.1011

with the use of multiple electrodes. These electrodes will be put on the patient's chest or wrist [7].

We need a system that can monitor sleep routines and process the signal data to measure sleep quality based on the patient's sleep cycle. The designed system will receive data from the sensor, process the data, determine the quality of the sleep, and visualize it within the application so it can be used by the doctor, health care, or close family to monitor and diagnose the patient's condition, helping doctor's decisions for the treatment. The concept that plays an essential part in this system is the Internet of Things (IoT), which connects electronics, sensors, and software that exchange data with each other [8]. Also, this system needs to be able to handle the incoming request when many users access the system. These requirements made the sleep monitoring system need to be scalable and available anytime.

The virtualization concept introduces a machine that has no physical attribute known as a Virtual Machine (VM), which simulates or virtualizes hardware component level making it possible to run multiple VM simultaneously as needed with current resources in a single machine. The problem is the VM needs pretty much cost maintaining performance and service uptime because the number of users increases along with requests per minute (rpm), making more resources to be added to guarantee the services can be used even in the peak condition, increasing the infrastructure costs [9]. The containerization concept has been proposed as a solution by doing virtualization on the Operating System level, which makes all dependencies, libraries, and binaries within the same container in the same machine, different from VM where each VM runs with its Operating System, making a container extremely lightweight and consume fewer resources compared to VM [10].

Wrapping each of our applications in the container will cause a challenge to manage this list of containers. Many container orchestrations tools have been developed in recent years to solve this challenge, such as Docker Swarm, Kubernetes, and Nomad, which helps to create automation for deploying, and provisioning. All of them have their advantages, for example, Docker Swarm claims that it has the fastest deployment time, Nomad can deploy non-container applications, and Kubernetes, which is the most used container orchestration has auto-scaling feature when there are big requests to the applications.

The main contributions of this paper are:

1) proposing an end-to-end sleep monitoring system architecture with high availability;

2) improving the scalability of applications by using Kubernetes as container orchestrations.

The rest of the paper is arranged as follows. Section 2 discusses the literature review. Section 3 discusses the proposed architecture and implementation. Finally, in Section 4, the results and analysis are presented and closed in Section 5 which presents the conclusion of the paper.

2. **Literature Review.**

Several methods have been proposed for the sleep monitoring system. Tal et al. [11] propose EarlySense, a contact-free sleep monitoring system to evaluate sleep stage and sleep quality based on ECG data signal. The ES contact-free sensor was placed under the mattress. The use of ECG as a base analysis is chosen because studies have demonstrated that ECG-based analysis produces an accurate distribution of the sleep stage. The ES sensor can be used for seamless monitoring without worrying about charging the sensor.

Lin et al. [12] propose SleepSense, a sleep monitoring system based on Doppler radar sensor that can differentiate or recognize various sleep status stages, including breathing section, bed exit, and on-bed movement. The SleepSense sleep stage classification can assess sleep quality, and the results can be used to know different sleep stages, sleep quality, and its disorders. The results of the study showed the SleepSense attains an overall 95.1% accuracy in identifying various sleep status and can be used in real life.

Utomo et al. [13] propose a sleep monitoring system based on IoT using an ECG sensor. Utomo et al. [13] use Weighted Extreme Learning Machine (WELM) with a combination with Particle Swarm Optimization (PSO) to overcome the imbalanced amount of dataset between sleep classes to analyze the sleep stage of the patient. The study uses the MIT-BIH Polysomnographic Database, which contains 10154 sleep stage annotated ECG. Utomo et al. [13] extract 18 total features from the annotated ECG and perform feature selection resulting in 10 final features. The results of the study show the proposed model acquired a mean accuracy of 78.78% for REM, NREM, and Wake stage and 73.09% for Light Sleep, Deep Sleep, REM, and Wake stage classification.

Surantha et al. [14] propose an IoT system for sleep quality monitoring using SCA11H, a contactless portable BCG sensor created by Murata Electronics Company, which solves the need for a regular sleep monitoring system. The sensor will record the heart and respiratory signals and send it to the data concentrator that is implemented on the Raspberry Pi 3 to do the data cleansing, extraction, and filtering based on B2B data. After that, the Raspberry Pi will send the pre-processed data to the cloud server using RESTful Web Service with post form via Wi-Fi (IEEE 802.11).

In the previous research, there is no study about sleep quality monitoring that considers dynamic change of the number of users, which usually occurs in real implementation. It can affect the scalability, availability, and performance of the monitoring system. Therefore, in this paper, we propose the sleep monitoring system based on container orchestration that can provide autoscaling, which is expected to deliver high performance with efficient use of resources.

3. Proposed Architecture and Implementation.

For this research, we will simulate the impact of container orchestration on the previous architecture of the sleep monitoring system, which has been done by Utomo et al. [13]. The system is being reused in this research with some rework and addition of some services to fit current research.

3.1. Sleep monitoring application architecture.

Sleep monitoring applications have three layers, which are sensor, cloud platform, and user interface. The sensors itself will send data to the cloud platform through message brokers such as Kafka, MQTT, and many others. This data will be received, processed, and stored on the cloud platform. For the user interface layer, users can preview the data through web and mobile applications, which will request the data from the cloud platform.

3.2. Sensor and data.

The device used in this study is MySignals Hardware (HW) paired up with Arduino Uno under the sensor. This device has built-in sensors that can be used for measuring different biometric measurements such as pulse, breath rate, blood oxygen level, ECG signals, blood pressure, and many others. ECG Sensor PRO for MySignals will work as the perceived layer that is used to get the ECG signal from the patient [15]. Later the Arduino Uno sent the ECG signal value to the Cloud Service using Python script via RESTful web service using a built-in ESP8266 Wi-Fi Module to connect to the Internet.

3.3. Applications flow.

Current architecture consists of three flows, which can be seen in Figure 2. The first flow starts with sensor gateway that receives data from the sensor and sends it to the message broker. The data persister will receive the message and save it into the database. After successfully saving the data, the data persister will push ECG value to the web socket. By creating a connection with a web socket, the mobile application will receive the data and visualize it within a graph. The second flow will be running when the scheduler is activated, which will trigger sleep classification and quantification services. On the classification service, the received ECG data will be used to get sleep quality using WELM-PSO proposed by Utomo et al. [13] and trigger the

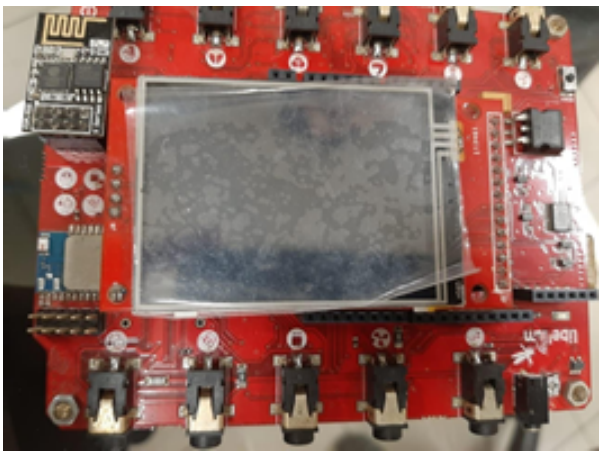

FIGURE 1. MySignals device module

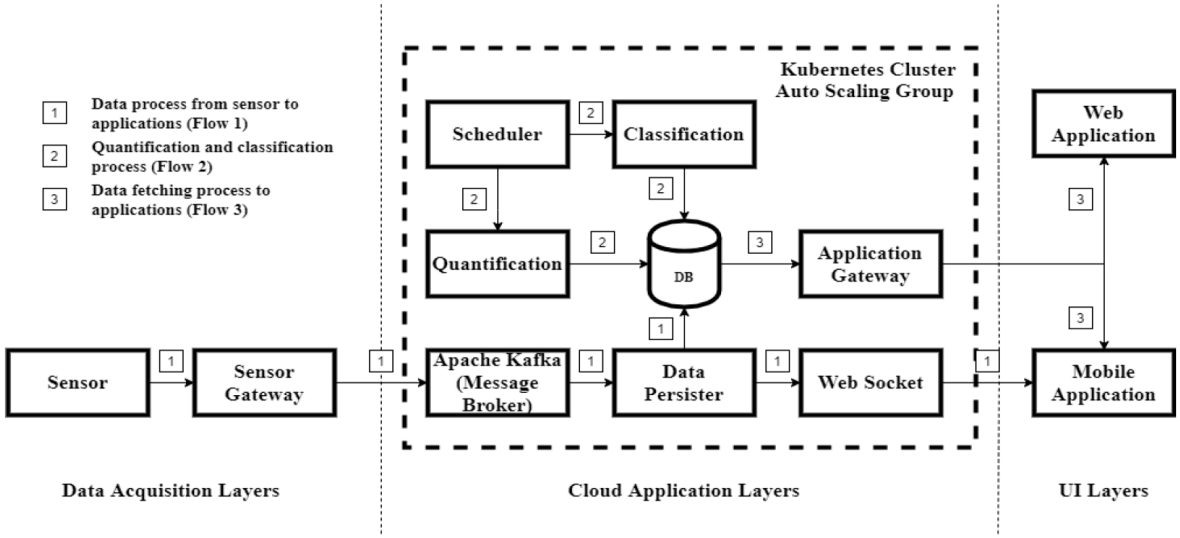

FIGURE 2. Sleep monitoring analysis architecture

quantification service. The quantification service will use the result from the classification service such as duration of a deep sleep, awake and total sleep time as input data for fuzzy logic made by Ang et al. to get sleep quality ranging from level 1 until level 9. The last flow is triggered when users access the web and mobile applications. This platform will get the processed data from classification and quantification services from the database through application gateway.

3.4. **Container orchestration implementation.** The proposed architecture wants every service in the sleep monitoring system to be containerized inside Docker containers and managed by container orchestrations. In this research, we propose Docker Swarm, Nomad, and Kubernetes to be tested as container orchestrations for the current system on Google Cloud Platform (GCP) as our cloud experiment environment. The configurations and software versions for the container orchestrations are shown in Table 1.

3.5. **Sleep monitoring web application.** The web application was developed using Spring Boot and included a feature to view ECG graphs from a person's sleep from the smartphone giving information about a person's sleep process. The web apps will generate an ECG graph based on the received ECG signal from the sensor, and the value is a real-time value obtained via socket application service. The ECG graph is shown in Figure 3 where the y-axis represents RR interval value and the x-axis represents time.

TABLE 1. Configuration of GCP virtual machine

Configurations	Value
Cloud platform	Google Cloud Platform
Image	Container-Optimized OS (cos)
Machine type	n1-standard-2
Number of CPU	2 vCPU
Memory size of each node	8 GB
Number of master nodes	1
Number of client nodes	2

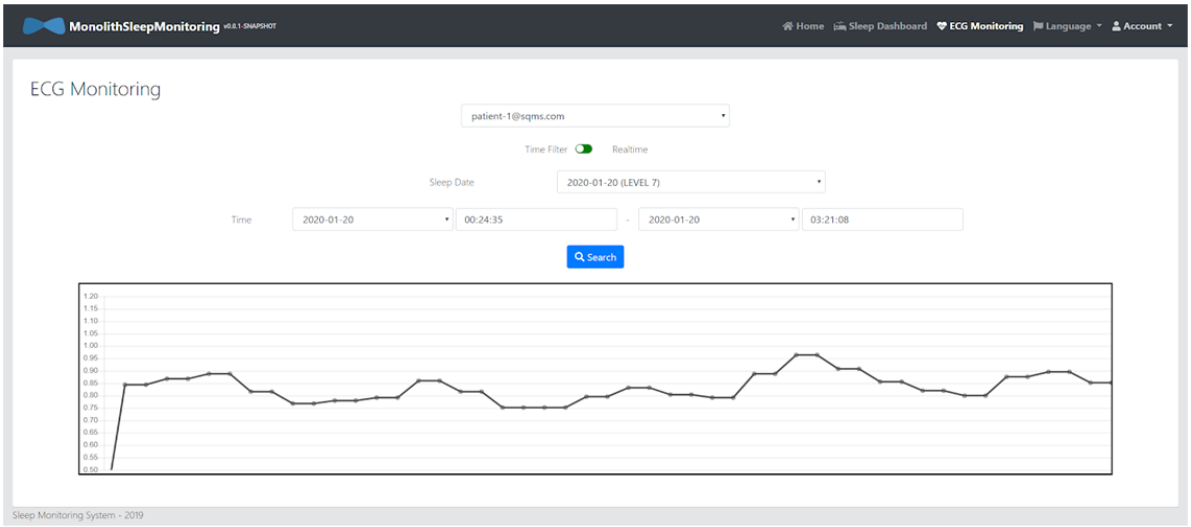

FIGURE 3. ECG monitoring on web application

4. **Evaluation Results.** In this research, we do not examine the classification model performance and only go to conduct load tests for each container orchestration with Apache JMeter. v.5.2.1. This classification model is just used to implement end-to-end architecture that we used in this research. Apache JMeter is an open-source Java-based application that is used for load testing, performance testing and stress testing to get the overall performance of the system [16]. Apache JMeter is used for many software systems or server types such as web servers, database servers, and others. The number of requests is going to be the testing parameters that start from 1000, 5000, and 10000 concurrent requests. From this testing, we are going to use two evaluation parameters: CPU utilization and successful request numbers. The testing scenario is shown in Figure 4.

4.1. **Successful request evaluation.** This evaluation is obtained whether the data which had been sent to the cloud platform is successfully saved on the cloud platform and sent to the web sockets. This result is illustrated in the last image of Figure 5. Each container orchestration has successfully handled all 1000 concurrent requests that have been sent on the first scenario. By increasing the number of requests, we can conclude that some failures are happening in the system. In the second scenario, which is 5000 concurrent requests, Kubernetes has 100% successful requests, Nomad has 89.84%, and docker swarm only has 88.62%. For the last scenario, Kubernetes is the best choice with 98.57%, followed by Nomad with 75.68%, and docker swarm with 47.52%. This evaluation shows that Kubernetes wins in every scenario which successfully handles most of the requests.

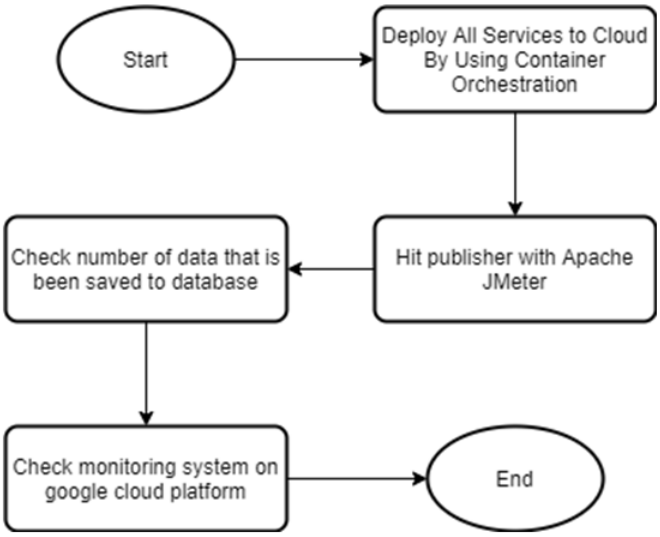

FIGURE 4. Testing scenario

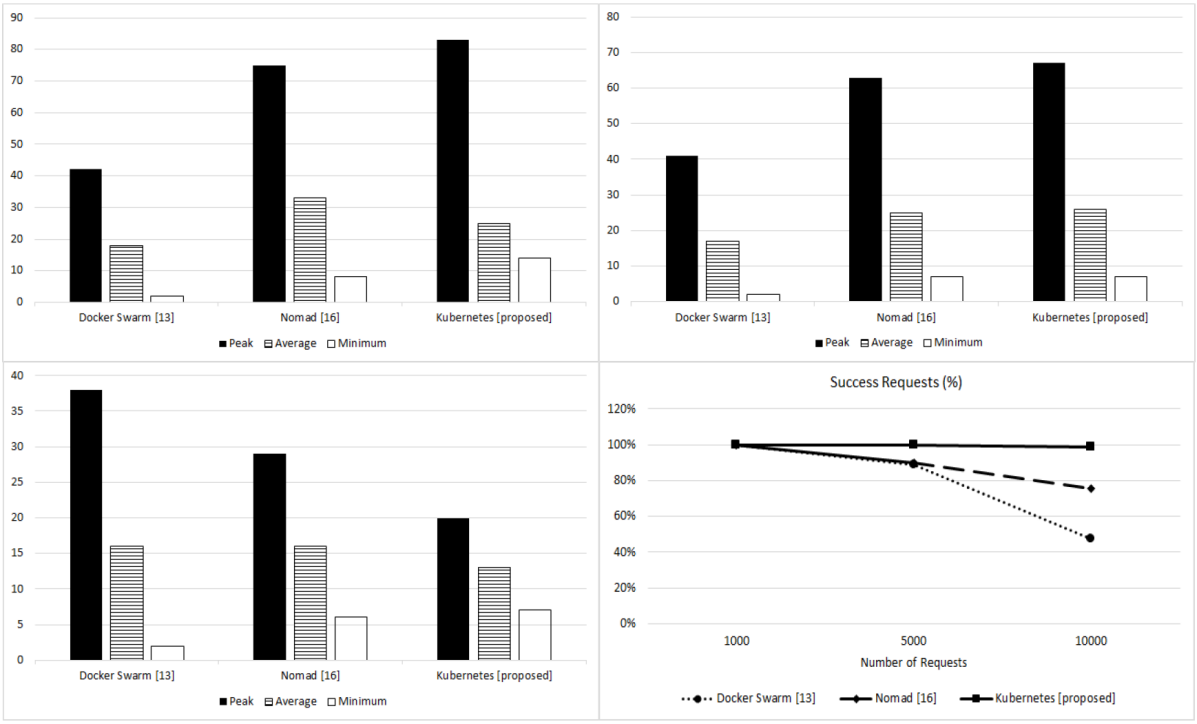

FIGURE 5. Evaluation of testing scenario with 1000, 5000 and 10000 concurrent requests

4.2. **CPU evaluation.** After simulating every testing scenario, we can inspect CPU utilization conditions when we are doing the load test on each container orchestration. There are three scenarios which are 1000, 5000 and 10000 concurrent request results. It is illustrated in Figure 5. This first scenario shows that Docker Swarm has the lowest CPU utilization in some states, which is 1.64% CPU utilization, but by average, it has the peak highest, which is 16.14% showing that Docker Swarm still has the lowest value of average, lowest, and peak highest CPU utilization, followed by Nomad, and the last is Kubernetes. However, Kubernetes held the highest success request rate, followed by Nomad and the Docker Swarm. This third scenario shows that Docker Swarm still has the lowest value of the average, lowest, and peak highest CPU utilization. Nomad comes in second with the exception of the average CPU utilization higher than Kubernetes. Furthermore, the

last one is Kubernetes, which has the highest value of both lowest and peak highest CPU utilization but has lower average CPU utilization compared to Nomad. Furthermore, the highest success request rate was held by Kubernetes, followed by Nomad and the Docker Swarm.

From the second and the third scenario, it can be deduced that although Docker Swarm always has the lowest CPU utilization but had the lowest success request rate compared to Nomad and Kubernetes. It is because the service is dead and will not respond to incoming requests. On the other hand, Kubernetes always had the highest success rate compared to the other two because of its autoscaling feature. With these results, we can conclude that Kubernetes utilize the resources that it has to ensure that the service can work properly, even in the heavy traffic.

5. **Conclusions.** In this study, we have proposed the deployment of the IoT system for sleep quality monitoring using Kubernetes container orchestration. The evaluation results show that Kubernetes, as the proposed container orchestration can achieve the lowest CPU utilization, error rate, and also network bandwidth compared to the other container orchestration such as Docker Swarm and Nomad. The obtained results can be achieved due to Kubernetes Load Balancer, which distributes the application traffic efficiently. Kubernetes also has the auto-scaling feature and cloud-based platform, making the configuration much easier compared to Docker Swarm and Nomad. For future study, it is recommended to evaluate these container orchestrations in another scenario, such as long term concurrent use and comparison with other container orchestration.

Acknowledgment. This work is supported by the Directorate General of Strengthening for Research and Development, Ministry of Research, Technology, and Higher Education, Republic of Indonesia as a part of Penelitian Terapan Unggulan Perguruan Tinggi Research Grant to Binus University entitled "Prototipe dan Aplikasi Monitoring Kualitas Tidur Portabel berbasis Teknologi Cloud Computing dan Machine Learning" or "Portable Sleep Quality Monitoring Prototype and Application based on Cloud Computing Technology and Machine Learning" with contract number: 039/VR.RTT/IV/2019 and contract date: 29 April 2019.

REFERENCES

[1] J. P. Chaput, C. Dutil and H. Sampasa-Kanyinga, Sleeping hours: What is the ideal number and how does age impact this?, *Nat. Sci. Sleep*, vol.10, pp.421-430, 2018.
[2] N. Surantha, G. P. Kusuma and S. M. Isa, Internet of Things for sleep quality monitoring system: A survey, *Proc. of the 11th Int. Conf. Knowledge, Inf. Creat. Support Syst*, 2017.
[3] R. Boostani, F. Karimzadeh and M. Nami, A comparative review on sleep stage classification methods in patients and healthy individuals, *Comput. Methods Programs Biomed.*, vol.140. pp.77-91, 2017.
[4] A. R. Hassan and A. Subasi, A decision support system for automated identification of sleep stages from single-channel EEG signals, *Knowledge-Based Systems*, vol.128, pp.115-124, 2017.
[5] J. Nedoma et al., Comparison of BCG, PCG and ECG signals in application of heart rate monitoring of the human body, *The 40th International Conference on Telecommunications and Signal Processing (TSP)*, Barcelona, Spain, pp.420-424, 2017.
[6] P. Dehkordi et al., Comparison of different methods for estimating cardiac timings: A comprehensive multimodal echocardiography investigation, *Front. Physiol.*, vol.10, 2019.
[7] J. M. Bishay and K. A. Wittman, *(12) United States Patent*, vol.2, no.12, 2017.
[8] M. Kumar, A review on applications of Internet of Things, *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol.7, no.6, pp.756-760, 2017.
[9] H. V. Netto, L. C. Lung, M. Correia, A. F. Luiz and L. M. Sá de Souza, State machine replication in containers managed by Kubernetes, *J. Syst. Archit.*, vol.73, pp.53-59, 2017.
[10] L. Herrera-Izquierdo and M. Grob, A performance evaluation between Docker container and Virtual Machines in cloud computing architectures, *Maskana*, no.12, pp.127-133, 2017.
[11] A. Tal, Z. Shinar, D. Shaki, S. Codish and A. Goldbart, Validation of contact-free sleep monitoring device with comparison to polysomnography, *J. Clin. Sleep Med.*, vol.13, no.3, pp.517-522, 2017.

[12] F. Lin et al., SleepSense: A noncontact and cost-effective sleep monitoring system, *IEEE Trans. Biomed. Circuits Syst.*, vol.11, no.1, pp.189-202, 2017.
[13] O. K. Utomo, N. Surantha, S. M. Isa and B. Soewito, Automatic sleep stage classification using weighted ELM and PSO on imbalanced data from single lead ECG, *Procedia Comput. Sci.*, vol.157, pp.321-328, 2019.
[14] N. Surantha, C. Adiwiputra, O. Kurniawan, S. Muhamad and B. Soewito, IoT system for sleep quality monitoring using ballistocardiography sensor, *Int. J. Adv. Comput. Sci. Appl.*, vol.11, no.1, pp.200-205, 2020.
[15] S.-C. Wang, Y.-L. Lin, M.-L. Chiang and H.-H. Pan, Improve the stability of the Internet of Things using dynamic load balancing clustering, *International Journal of Innovative Computing, Information and Control*, vol.16, no.1, pp.63-76, 2020.
[16] N. Jha and R. Popli, Comparative analysis of web applications using JMeter., *Int. J. Adv. Res. Comput. Sci.*, vol.8, no.3, pp.774-777, 2017.