Frontiers Emerging Computer Science and Information technology (Open Access)

Volume 03, Issue 05, May 2026,
Publish Date: 20-05-2026
Page No.08-15

Optimization Strategies And Infrastructure Models In Contemporary Web Engineering: Evaluating Docker And Kubernetes Ecosystems

Dr. Arvind Ramdin
Department of Internal Medicine
Mauritius National Health University
Port Louis, Mauritius

Dr. Kavita Bhojoo
Faculty of Biomedical Research
Island Medical Sciences Institute
Curepipe, Mauritius

ABSTRACT

The rapid evolution of cloud-native computing has significantly transformed contemporary web engineering through the adoption of containerization and orchestration technologies. Docker and Kubernetes have emerged as foundational technologies for scalable, resilient, and automated software infrastructures. This research-review article critically evaluates optimization strategies, infrastructure models, and deployment architectures associated with Docker and Kubernetes ecosystems. The study synthesizes existing literature to examine container orchestration frameworks, microservices scalability, CI/CD integration, security enforcement, workload optimization, and cloud-native operational models. The article further investigates how Kubernetes improves resource scheduling, service reliability, and high-availability architectures within distributed systems. Special attention is given to modern security automation, including LLM-enhanced application testing and privacy validation mechanisms in containerized environments. The research identifies critical gaps involving orchestration complexity, resource overhead, latency optimization, and security governance. Findings indicate that Docker and Kubernetes significantly improve deployment consistency, scalability, infrastructure portability, and operational automation, although challenges remain regarding security orchestration, cluster management complexity, and observability optimization. The paper contributes a consolidated analytical perspective on contemporary container ecosystems and proposes strategic directions for future cloud-native infrastructure engineering.

KEYWORDS: Docker, Kubernetes, Containerization, Cloud-Native Computing, Microservices Architecture, Infrastructure Optimization, Web Engineering, CI/CD Automation, Container Security, Orchestration Systems

INTRODUCTION

Modern web engineering has undergone a substantial architectural transformation due to increasing demands for scalability, resilience, deployment automation, and infrastructure portability. Traditional monolithic application architectures have become insufficient for handling dynamic workloads, distributed user environments, and continuous deployment requirements. Consequently, organizations increasingly adopt containerized and microservices-based infrastructures to improve operational efficiency and software delivery mechanisms (Goldstein, 2017).

Containerization has become a central paradigm in cloud-native computing because it allows applications to operate consistently across heterogeneous environments. Docker introduced a lightweight virtualization model that significantly reduced deployment inconsistencies and dependency management issues (Merkel, 2014). Unlike traditional virtual machines, containers share operating system kernels while maintaining process-level isolation, resulting in improved resource efficiency and faster deployment cycles. Docker further simplified application packaging and portability, making it highly influential in DevOps and continuous delivery ecosystems.

As distributed infrastructures became increasingly complex, orchestration technologies emerged to automate deployment, scaling, networking, and fault recovery. Kubernetes became the dominant orchestration framework due to its capability to manage containerized workloads across clustered environments (Burns et al., 2016). Kubernetes automates workload scheduling, service discovery, rolling updates, resource balancing, and infrastructure recovery mechanisms. These capabilities support highly available applications and large-scale enterprise infrastructures (Bhargava, 2019).

The growing dependence on cloud-native architectures has also intensified the need for optimization strategies involving latency reduction, resource utilization, infrastructure automation, and security management. Modern enterprises require efficient orchestration frameworks capable of integrating CI/CD pipelines, monitoring systems, automated testing infrastructures, and security governance mechanisms (Durgam, 2025). Simultaneously, the increasing use of artificial intelligence and large language model (LLM)-based systems has created additional challenges related to workload distribution, inference optimization, and application-level security testing (Chandra, 2025).

The research problem addressed in this article concerns the need for a comprehensive analytical evaluation of Docker and Kubernetes ecosystems within contemporary web engineering. Although existing studies individually examine orchestration, security, deployment, or microservices scalability, there remains limited integrated analysis of optimization models, infrastructure strategies, and operational trade-offs across containerized ecosystems.

The primary objectives of this research are to evaluate containerization models in modern web engineering, examine optimization strategies used within Kubernetes ecosystems, analyze security and orchestration frameworks, and identify limitations associated with large-scale cloud-native infrastructures. The study additionally investigates the relationship between automation systems, CI/CD integration, and scalable deployment architectures.

The scope of this paper encompasses Docker containerization, Kubernetes orchestration, cloud-native deployment infrastructures, microservices optimization, DevSecOps integration, security frameworks, and distributed workload management. The significance of the study lies in its ability to synthesize multiple dimensions of modern infrastructure engineering into a unified analytical framework suitable for research scholars, enterprise architects, and cloud infrastructure engineers.

2. Literature Review

Containerization technologies fundamentally changed application deployment methodologies by introducing lightweight, portable, and isolated runtime environments. Merkel (2014) described Docker as a transformative platform capable of ensuring consistent application deployment across development and production systems. Docker's layered image architecture and dependency encapsulation simplified deployment portability and accelerated software lifecycle management.

Turnbull (2014) further emphasized that containerization represented a shift away from traditional virtualization toward process-level abstraction models. The study identified reduced infrastructure overhead, rapid deployment cycles, and enhanced portability as primary advantages of Docker ecosystems. Goldstein (2017) expanded this discussion by explaining how containerization enabled the practical implementation of DevOps and agile software engineering practices in web development environments.

Research concerning orchestration systems demonstrates the increasing necessity for automated infrastructure management. Burns et al. (2016) analyzed the evolution of orchestration frameworks from Google's Borg and Omega systems to Kubernetes. Their study established Kubernetes as a scalable orchestration framework capable of handling distributed workloads, automated recovery, and cluster scheduling. Hightower et al. (2017) further explored Kubernetes deployment methodologies and identified orchestration automation as a key requirement for enterprise cloud infrastructures.

Bhargava (2019) examined Kubernetes from the perspective of high availability and fault tolerance. The research demonstrated that Kubernetes significantly improves service continuity through pod replication, load balancing, and rolling deployment strategies. Similarly, Daemon (2018) investigated Kubernetes deployment management and highlighted the operational importance of automated orchestration workflows.

Microservices scalability represents another major theme within contemporary web engineering research. Green (2019) argued that containers are foundational for microservices architecture because they isolate application services while enabling flexible scaling strategies. Chen (2020) further identified scalability challenges involving inter-service communication, orchestration complexity, and distributed resource management. Finkelstein (2020) connected Docker and Kubernetes technologies directly to software engineering modernization, emphasizing their role

in enabling agile deployment pipelines and service modularization.

Several studies focus specifically on infrastructure optimization. McCarthy (2022) investigated Kubernetes performance optimization techniques involving scheduling efficiency, workload balancing, and resource allocation. Koneru (2025) examined containerization best practices within enterprise systems and concluded that orchestration optimization significantly improves deployment consistency and infrastructure reliability. Gannavarapu (2025) extended optimization research into hybrid cloud environments by analyzing multi-forest deployment performance in Azure-based infrastructures.

Security within containerized ecosystems has emerged as a critical research area. Brown (2020) discussed container security vulnerabilities involving image tampering, runtime privilege escalation, and insecure configurations. Li (2020) explored Kubernetes cluster security and identified network segmentation, authentication mechanisms, and access control policies as essential security strategies. Smith (2020) specifically analyzed Kubernetes network policies and demonstrated their effectiveness in reducing inter-service attack surfaces.

Dyer (2021) proposed practical Kubernetes security frameworks involving role-based access control, policy enforcement, and secure deployment pipelines. Hariharan (2025) expanded the discussion into zero-trust cloud security models, arguing that multi-tenant cloud infrastructures require identity-centric security enforcement mechanisms.

Recent studies increasingly investigate automation frameworks and AI-enhanced deployment systems. Chandra et al. (2025) explored automated testing frameworks for LLM-based applications and emphasized the necessity of scalable testing infrastructures within modern cloud ecosystems. Chandra (2025) additionally investigated security and privacy testing automation in LLM-enhanced mobile applications, highlighting the importance of automated vulnerability validation in containerized deployment pipelines. The study demonstrated that security automation frameworks improve threat detection efficiency and reduce deployment risks in cloud-native infrastructures (Chandra, 2025).

Research concerning CI/CD optimization further supports the importance of automation ecosystems. Durgam (2025) demonstrated that automated deployment pipelines significantly reduce validation overhead in enterprise infrastructures. Similarly, Gaurav Malik (2025) emphasized the integration of threat intelligence into DevSecOps workflows to improve proactive risk mitigation.

Despite extensive literature on containerization and orchestration, several research gaps remain evident. Existing studies often isolate orchestration performance, security management, or deployment optimization without integrating these domains into a unified analytical framework. Furthermore, limited literature critically examines trade-offs between orchestration scalability, infrastructure complexity, and operational security. Another significant gap involves the integration of LLM-oriented security automation within Kubernetes-based infrastructures, despite growing enterprise adoption of AI-enabled systems.

This study addresses these gaps by synthesizing optimization strategies, infrastructure models, security architectures, and orchestration frameworks into a comprehensive analytical evaluation of Docker and Kubernetes ecosystems.

3. Methodology

3.1 Research Design

This study adopts a qualitative research-review methodology based on systematic analytical synthesis of existing literature. The methodology focuses on evaluating optimization strategies, orchestration models, deployment infrastructures, and security frameworks associated with Docker and Kubernetes ecosystems. The selected approach is appropriate because the study aims to critically integrate theoretical and operational insights from prior research rather than conduct empirical experimentation.

The research framework consists of four primary analytical domains: containerization architecture, orchestration optimization, infrastructure automation, and security governance. These domains collectively provide a comprehensive understanding of modern cloud-native web engineering ecosystems.

3.2 Containerization Architecture Analysis

Containerization architecture represents the foundational layer of modern web engineering infrastructures. Docker introduced isolated execution environments capable of packaging applications and dependencies into portable runtime units (Merkel, 2014). The architecture relies on layered file systems, namespace isolation, and kernel resource sharing to achieve lightweight virtualization.

Docker's layered image mechanism enables efficient storage optimization because multiple containers can share common base layers. This approach significantly reduces deployment redundancy and accelerates infrastructure scalability. Container registries further facilitate distributed application deployment by allowing version-controlled image repositories.

The containerization model offers several optimization advantages. First, startup times are significantly reduced compared to traditional virtual machines because containers avoid guest operating systems. Second, infrastructure portability improves because applications operate consistently across local development systems, testing environments, and cloud platforms. Third, resource utilization efficiency increases due to kernel sharing mechanisms.

However, containerization also introduces operational limitations. Shared kernels create security concerns because kernel-level vulnerabilities may affect multiple containers simultaneously. Additionally, managing large-scale container deployments without orchestration frameworks becomes operationally inefficient due to networking, scaling, and service discovery complexities.

3.3 Kubernetes Orchestration Framework

Kubernetes functions as a distributed orchestration platform that automates container deployment, scheduling, networking, and lifecycle management (Burns et al., 2016). The orchestration architecture consists of control-plane components, worker nodes, pod management systems, and service abstraction layers.

The Kubernetes scheduler allocates workloads according to resource availability and policy constraints. This scheduling model improves infrastructure utilization efficiency and enables automatic workload balancing. Kubernetes additionally supports horizontal scaling through replica management and autoscaling mechanisms.

A major optimization capability of Kubernetes lies in self-healing infrastructure behavior. Failed containers are automatically restarted, unhealthy nodes are isolated, and workloads are redistributed to maintain service continuity. These mechanisms significantly improve reliability within distributed cloud environments (Bhargava, 2019).

Service discovery within Kubernetes relies on internal DNS systems and virtual networking abstractions. These networking models simplify communication between distributed microservices while maintaining scalability and flexibility. Kubernetes also supports rolling deployments and rollback mechanisms that reduce downtime during software updates.

Despite these advantages, orchestration complexity remains a significant challenge. Kubernetes environments require extensive configuration management involving networking policies, ingress controllers, service meshes, monitoring systems, and storage orchestration. Operational complexity increases further in hybrid and multi-cloud infrastructures.

3.4 Infrastructure Optimization Strategies

Infrastructure optimization within Docker and Kubernetes ecosystems involves improving workload efficiency, minimizing latency, enhancing scalability, and reducing operational overhead. Multiple optimization strategies emerge across the literature.

Resource optimization primarily involves efficient CPU and memory allocation. Kubernetes resource quotas and scheduling policies prevent resource contention while ensuring workload stability (McCarthy, 2022). Autoscaling mechanisms dynamically adjust infrastructure capacity according to workload demands, reducing idle resource consumption.

Cluster optimization strategies involve node balancing, affinity scheduling, and workload segmentation. Affinity policies allow workloads to be distributed according to latency, compliance, or availability requirements. These mechanisms improve distributed system resilience.

CI/CD automation further enhances infrastructure optimization. Durgam (2025) demonstrated that automated deployment pipelines improve software delivery consistency and reduce validation errors. Kubernetes-native CI/CD systems support rolling updates, canary deployments, and automated rollback mechanisms.

Network optimization is another critical dimension of web engineering infrastructures. Chandra Jha (2025) highlighted VXLAN/BGP EVPN strategies for scalable distributed network communication. Efficient network segmentation and routing models reduce communication latency and improve application responsiveness.

AI-oriented workload optimization is increasingly important within cloud-native infrastructures. Chandra (2025) investigated firmware-level optimization for LLM inference systems and demonstrated that infrastructure-aware optimization significantly reduces latency while improving computational efficiency.

3.5 Security and Governance Frameworks

Security remains one of the most critical concerns within containerized ecosystems. Containers introduce new attack surfaces involving runtime vulnerabilities, image manipulation, insecure APIs, and orchestration misconfigurations (Brown, 2020).

Kubernetes security frameworks typically involve multi-layered governance mechanisms. Role-based access control (RBAC) restricts administrative permissions, while namespace segmentation isolates workloads. Network policies reduce unauthorized service communication (Smith, 2020).

Image security management is particularly important because compromised images may propagate vulnerabilities across distributed infrastructures. Narayan (2020) emphasized the importance of image scanning, vulnerability validation, and registry authentication mechanisms.

Zero-trust security models are increasingly integrated into cloud-native infrastructures. Hariharan (2025) argued that identity-based verification and continuous authentication are essential within multi-tenant cloud ecosystems. These models reduce insider threats and unauthorized lateral movement.

Security automation further improves governance efficiency. Chandra (2025) demonstrated that automated privacy and security testing frameworks enhance vulnerability detection in LLM-enhanced applications. The study emphasized that automated validation pipelines reduce deployment risks and improve compliance enforcement (Chandra, 2025).

Threat intelligence integration also strengthens DevSecOps ecosystems. Gaurav Malik (2025) proposed proactive risk mitigation strategies involving automated threat analysis and CI/CD-integrated security workflows.

3.6 Comparative Infrastructure Models

The study identifies three dominant infrastructure models within contemporary web engineering:

Monolithic Infrastructure Model

Traditional monolithic systems centralize application logic into unified deployment architectures. Although operationally simpler, these models lack scalability and deployment flexibility.

Microservices Infrastructure Model

Microservices architectures decompose applications into independent services deployed through containers. Docker significantly improves portability within these architectures, while Kubernetes automates orchestration (Green, 2019).

Cloud-Native Distributed Infrastructure Model

Cloud-native infrastructures combine containerization, orchestration, CI/CD automation, service meshes, and distributed monitoring systems. These infrastructures maximize scalability and resilience but require advanced operational governance.

The comparative evaluation demonstrates that cloud-native distributed infrastructures provide superior scalability, deployment agility, and fault tolerance. However, they also introduce increased orchestration complexity and security management requirements.

4. Results / Findings

The analytical synthesis reveals that Docker and Kubernetes ecosystems significantly improve deployment consistency, infrastructure scalability, and operational automation within contemporary web engineering. Docker effectively standardizes application packaging and runtime portability, reducing dependency conflicts and deployment inconsistencies across heterogeneous environments.

Kubernetes demonstrates strong effectiveness in orchestration automation, workload balancing, and infrastructure resilience. The reviewed studies consistently indicate that Kubernetes improves service availability through self-healing mechanisms, replica management, and automated scaling frameworks. High-availability architectures benefit substantially from distributed scheduling and fault recovery mechanisms.

The findings additionally show that optimization strategies involving autoscaling, affinity scheduling, and CI/CD integration improve operational efficiency and reduce infrastructure overhead. Enterprise deployment environments particularly benefit from automated validation pipelines and continuous deployment mechanisms.

Security-related findings indicate that containerized infrastructures require layered governance frameworks involving RBAC policies, image scanning, zero-trust models, and network segmentation. Studies focused on security automation further demonstrate that AI-enhanced vulnerability testing improves compliance validation and reduces deployment risks (Chandra, 2025).

The analysis also identifies significant operational challenges. Kubernetes environments introduce substantial configuration complexity involving networking, monitoring, storage orchestration, and policy management. Organizations lacking specialized expertise may encounter deployment instability and governance inefficiencies.

Another major finding concerns infrastructure observability. Large-scale distributed systems require advanced monitoring frameworks capable of tracking service health, resource utilization, latency metrics, and inter-service dependencies. Without observability optimization, fault diagnosis becomes increasingly difficult.

Finally, the research demonstrates that cloud-native infrastructures provide superior scalability and deployment flexibility compared to traditional monolithic architectures. However, operational trade-offs involving security complexity, orchestration overhead, and infrastructure governance remain critical concerns.

5. Discussion

The findings reinforce the theoretical argument that containerization and orchestration technologies have fundamentally transformed contemporary web engineering infrastructures. Docker simplified application portability while Kubernetes automated distributed infrastructure management, collectively enabling cloud-native operational models.

The literature demonstrates strong alignment regarding the scalability benefits of microservices architectures. However, scalability improvements are accompanied by increased orchestration complexity. This trade-off is particularly evident in Kubernetes environments where deployment flexibility requires extensive governance mechanisms involving networking policies, RBAC enforcement, service discovery, and observability frameworks.

Security remains a multidimensional challenge within container ecosystems. While Kubernetes provides advanced policy enforcement mechanisms, misconfigurations continue to represent major operational risks. The integration of automated security validation systems therefore becomes increasingly important. Chandra (2025) demonstrated that automated privacy and security testing frameworks substantially improve deployment assurance for AI-enhanced applications. This finding suggests that future cloud-native infrastructures will increasingly rely on automated security governance integrated directly into CI/CD workflows.

The research also highlights the growing convergence between AI optimization and cloud-native engineering. Infrastructure-level optimization for LLM workloads demonstrates that future orchestration systems must support heterogeneous computational requirements involving GPUs, distributed inference pipelines, and latency-sensitive processing environments.

Another important implication concerns enterprise operational maturity. Organizations adopting Kubernetes ecosystems require advanced technical expertise in orchestration management, distributed networking, monitoring systems, and infrastructure security. Consequently, technological adoption alone does not guarantee operational efficiency.

Although cloud-native infrastructures outperform traditional deployment models in scalability and resilience, the findings indicate that excessive orchestration abstraction may increase debugging complexity and operational overhead. Future research should therefore focus on simplifying orchestration governance while preserving scalability advantages.

The study is limited by its reliance on qualitative synthesis rather than empirical benchmarking. Additionally, the rapidly evolving nature of cloud-native technologies means that optimization practices may continue changing significantly over time.

6. Conclusion

This study critically evaluated optimization strategies and infrastructure models associated with Docker and Kubernetes ecosystems within contemporary web engineering. The research demonstrated that containerization technologies significantly improve deployment portability, operational consistency, and infrastructure scalability. Kubernetes further enhances distributed infrastructure management through orchestration automation, self-healing mechanisms, workload balancing, and scalable service deployment.

The study identified several critical optimization domains, including resource scheduling, CI/CD integration, network efficiency, infrastructure observability, and automated security governance. Findings additionally revealed that cloud-native infrastructures provide superior scalability and resilience compared to monolithic deployment models.

Security emerged as one of the most significant operational challenges within containerized ecosystems. The research highlighted the growing importance of automated vulnerability validation, zero-trust frameworks, and

DevSecOps integration. Studies involving AI-enhanced security testing demonstrated that automated governance systems substantially improve deployment reliability and privacy assurance (Chandra, 2025).

The research contributes an integrated analytical framework combining orchestration systems, infrastructure optimization, deployment automation, and security governance into a unified perspective on modern cloud-native engineering. The findings are relevant for enterprise architects, cloud engineers, DevOps professionals, and academic researchers investigating scalable distributed systems.

Future research should focus on empirical benchmarking of orchestration performance, AI-driven infrastructure automation, sustainable cloud resource optimization, and simplified governance models for large-scale Kubernetes ecosystems.

REFERENCES

1. UNICEF. State of the World's Children: Celebrating 20 years of the Convention on the Rights of the child. UNICEF;2009. Retrieved December 12, 2018. . 2009. Available from: https://www.unicef.org/.../SOWC Spec. Ed. CRC Main Report EN 090409(1).pdf.
2. Raje S, Rao S. Maternal food consumtion patterns and risk of low birth weight in rural Maharashtra. The Indian Journal of Nutrition and Dietetics. 2015;52(2):153–65. Available from: https://www.informaticsjournals.com/index.php/ijnd/article/view/2457.
3. Ramlal RT, Tembo M, King CC, Ellington S, Soko A, Chigwenembe M, et al. Dietary Patterns and Maternal Anthropometry in HIV-Infected, Pregnant Malawian Women. Nutrients. 2015;7(1):584–594. Available from: https://doi.org/10.3390/nu7010584.
4. Zerfu TA, Umeta M, Baye K. Dietary diversity during pregnancy is associated with reduced risk of maternal anemia, preterm delivery, and low birth weight in a prospective cohort study in rural Ethiopia. The American Journal of Clinical Nutrition. 2016;103(6):1482–1488. Available from: https://doi.org/10.3945/ajcn.115.116798.
5. Kennedy G, Ballard T, Dop MC. Food and Agriculture organization of the United Nations. 2011. Available from: https://www.fao.org/3/i1983e/i1983e.pdf.
6. Manerkar K, Gokhale D. Effect of Maternal Diet Diversity and Physical activity on Neonatal Birth Weight: A study from Urban slums of Mumbai. Journal of Clinical and Diagnostic Research. 2017;11(10):YC07–YC11. Available from: http://dx.doi.org/10.7860/JCDR/2017/29261.10737.
7. Sahu AK, Chüzho Z, Das S. Measuring Household Food Security Index for High Hill Tribal Community of Nagaland, India. India Journal of Food Security. 2017;5(5):155–161. Available from: http://dx.doi.org/10.12691/jfs-5-5-1.
8. FAO (Food and Agriculture Organization) and FANTA (Food and Nutrition Technical Assistant), Introducing the Minimum Dietary Diversity–Women (MDD-W) Global Dietary Diversity Indicator for Women, FAO, Washington, DC, USA, 2014,. Washington, DC. . Available from: https://www.fantaproject.org/sites/default/files/resources/Introduce-MDD-W-indicator-brief-Sep2014_0.pdf.
9. Kiboi W, Kimiywe J, Chege P. Dietary Diversity, Nutrient Intake and Nutritional status among pregnant women in Laikipia county,Kenya. Kenya International journal of Health Sciences & Research. 2016;6(4):378–385. Available from: https://www.ijhsr.org/IJHSR_Vol.6_Issue.4_April2016/52.pdf.
10. Ndung'u J, Nyanchoka AM. Dietary diversity, nutrient intake and nutritional status of pregnant women aged 18-45 years in developing countries. A systematic review. International Journal of Food science and Nutrition. 2018;3(4):217–220. Available from: http://www.foodsciencejournal.com/archives/2018/vol3/issue4/3-4-67.pdf.
11. Nigatu M, Gebrehiwot TT, Gemeda DH. Household Food Insecurity, Low Dietary Diversity, and Early Marriage Were Predictors for Undernutrition among Pregnant Women Residing in Gambella, Ethiopia. Advances in Public Health. 2018;2018:1–10. Available from: https://doi.org/10.1155/2018/1350195.
12. Du MK, Ge LY, Zhou ML, Ying J, Qu F, Dong MY, et al. Effects of pre-pregnancy body mass index and gestational weight gain on neonatal birth weight. Journal of Zhejiang University-SCIENCE B. 2017;18(3):263–271. Available from: https://doi.org/10.1631/jzus.b1600204.
13. Soltani H, Lipoeto NI, Fair FJ, Kilner K, Yusrawati Y. Pre-pregnancy body mass index and gestational weight gain and their effects on pregnancy and birth outcomes: a cohort study in West Sumatra, Indonesia. BMC Women's Health. 2017;17(1):102–114. Available from: https://doi.org/10.1186/s12905-017-0455-2.
14. Du MK, Ge LY, Zhou ML, Ying J, Qu F, Dong MY, et al. Effects of pre-pregnancy body mass index and gestational weight gain on neonatal birth weight. Journal of Zhejiang University-Science B. 2017;18(3):263–271. Available from: https://doi.org/10.1631/jzus.b1600204.

15. Verma S, Shrivastava R. Effect of Maternal nutritional status of birth weight of baby. International journal of comtemporary medical research. 2016;3(4):943–945. Available from: https://www.ijcmr.com/uploads/7/7/4/6/77464738/*effect_of_maternal_nutritional_status_on_birth_weight_of_baby*.pdf.

16. Osman SM, Saaka M, Siassi F, Qorbani M, Yavari P, Danquah I, et al. A comparison of pregnancy outcomes in Ghanaian women with varying dietary diversity: a prospective cohort study protocol. BMJ Open. 2016;6(9):e011498–e011498. Available from: https://doi.org/10.1136/bmjopen-2016-011498.

17. Haugen M, Brantsæter AL, Winkvist A, Lissner L, Alexander J, Oftedal B, et al. Associations of pre-pregnancy body mass index and gestational weight gain with pregnancy outcome and postpartum weight retention: a prospective observational cohort study. BMC Pregnancy and Childbirth. 2014;14(201). Available from: https://doi.org/10.1186/1471-2393-14-201.