Mathematical Statistician and Engineering Applications
ISSN: 2094-0343
2326-9865

Comparison between Headless CMS and Backend-as-a-Service Products for E-Suripreneur Backend

Nur Ayuni Nor Sobri[1], Mohamad Aqib Haqmi Abas[1], Ahmad Ihsan Mohd Yassin[2*], Megat Syahirul Amin Megat Ali[2], Nooritawati Md Tahir[1,3], Azlee Zabidi[4], Zairi Ismael Rizman[5]

[1]School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia

[2]Microwave Research Institute, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia

[3]Institute for Big Data Analytics and Artificial IntelligencE (IBDAAI), Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia

[4]Faculty of Systems & Software Engineering, College of Computing & Applied Sciences, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia

[5]School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, 23000 Dungun, Terengganu, Malaysia

Corresponding author: ihsan.yassin@gmail.com

Article Info
Page Number: 928 – 938
Publication Issue:
Vol. 71 No. 3s2 (2022)

Article History
Article Received: 28 April 2022
Revised: 15 May 2022
Accepted: 20 June 2022
Publication: 21 July 2022

Abstract

Numerous businesses have entered the business market in recent years, each focusing on a specialised sector or industry. They often have a trait that distinguishes them from the competition. The major goal of startups is to swiftly bring their product to market and build user momentum for their product and system. However, there might be various problems and roadblocks to bringing the product to market, one of which being the product's software development. Fortunately, there are various development tools available today to assist with bootstrapping the early complexity of development, which is beneficial for companies trying to swiftly bring their product to market. Backend-as-a-service (BaaS) and headless content management systems (CMS) are two solutions that aid in simplifying and increasing the pace of software development, allowing for a faster time to market. This article compares Strapi, Directus, and Supabase as possible backend systems for the e-Suripreneur system. Directus appears to be the best fit for the e-Suripreneur system, as it meets the system's baseline requirements and provides various additional benefits over Strapi and Supabase.

Keywords: headless content management system (CMS), backend as a service (BaaS), backend application system, business startup.

Introduction:

Multiple software technology companies have entered the market in the last few years with products that are aimed at a variety of business sectors and industries. These include software businesses focused on e-hailing, online meal delivery, e-commerce and others. When a

software startup company or similar joins the commercial world, they typically have a distinguishing feature that sets them apart from the competition. Their primary objective is to get their product to market quickly and gain user momentum for their product and system [1-3]. Nowadays, there are numerous development tools available to assist with bootstrapping the early complexity of development, which is advantageous for startups looking to quickly get their product to market.

Backend-as-a-service (BaaS) and headless content management systems (CMS) are two of the most popular software development tools for this. These two technologies make it simple to set up the backend of an application and typically work by sending data via an Application Programming Interface (API). Thus, developers can concentrate exclusively on the frontend and design; it makes no difference whether the frontend is web-based or mobile-based; the only requirement is for the application to connect to the backend system via API to fetch and send data. Having a backend-as-a-service is essential nowadays since most of the repetitive effort and boilerplate code associated with creating, reading, updating, and deleting data can be eliminated, drastically lowering development costs.

Backend as a Service (BaaS) is a cloud service model that enables mobile and online apps to be connected to cloud-based backend services [4-8]. Data storage, user management, file storage, geolocation, push alerts, social integration, and analytics are just few of the services that fall under this category [5]. BaaS platforms, at their core, let you store your data without the hassles and costs associated with establishing and maintaining distinct services for each application [9]. By connecting to an Application Programming Interface (API) or Software Development Kit (SDK), BaaS enables developers to focus on application features rather than backend development. It facilitates and accelerates the software development process by providing the backend for apps. The BaaS market has risen significantly over the last decade, as an increasing number of developers have embraced BaaS services. Rapid adoption and development is a crucial factor driving the BaaS market's growth. The primary benefits of BaaS include accelerated time to market, reduced development costs, and enhanced scalability.

A content management system (CMS) is a software application that is used to create, edit, and manage content [10-16]. It enables users to manage material from a centralised location [12, 17-20]. Additionally, a web-based CMS enables the creation of websites that can be easily updated by non-technical staff members. Although the term may refer to manual content management processes, it is primarily used for a variety of software solutions that enable advanced management of large amounts of information due to their strength, flexibility, and extendability. It can be used to sync data from multiple sources, execute collaborative projects, and organise work in corporate settings [11]. Additionally, it has been widely used in commercial, media, financial, and social applications [21].

Headless Content Management Systems (CMS) reimagine Content Management Systems. Headless CMS stores and publishes content via API connectivity and backend technologies [22, 23]. Headless CMS enables storage and dissemination of content to be managed by other apps. A headless CMS's major objective is to decouple content creation, storage, and administration from display and delivery. This covers concerns about security, scalability,

usability, and frontend technologies. With the separation of the backend and frontend, content and design may be more adaptable [22, 23]. In comparison to typical content management systems, where only template designs are offered.

We conduct a thorough analysis of two headless CMS (Strapi [24] and Directus [25]) and a BaaS (Supabase [26]) application that will be used to manage the backend of the e-Suripreneur system in this paper. The application must meet the requirements of the e-Suripreneur system, which include a basic create, read, update, and delete (CRUD) application, user authentication and authorization, database migration capabilities, inexpensive and have the ability to self-host the application in own server, the ability to customise by adding custom code to handle more complex business logic, and the ability to send data to the client side for end users via the Representational State Transfer Application Programming Interface (REST API) or another widely supported method as well as its own Software Development Kit (SDK).

Related Works:

Numerous articles have been discovered revolving around the same study's theme, such as [23], in which the author developed a new website for the Swiss-Hema federation utilising Strapi headless CMS as the application system's backend. Strapi was chosen because it met the client's criteria for a CMS with an easy-to-use interface for users with no prior understanding of information technology (IT), support for internationalisation (i18n), and event registration with a payment mechanism. The author chose headless CMS technology over a standard CMS for the project due to the headless CMS's flexibility and customization features. The author can customise the web application's frontend, which results in a solution that is better fitted to the client's demands and more stable than an assembly of numerous separate functionalities as there is with standard CMS.

The author of [27] develops a web application that is compliant with web design best practises, user-friendly, inexpensive, and maintainable with little IT expertise for Vihdin-Nummelan Kylähistoria Ry, a non-profit organisation. The development work was carried out in an agile manner, using the Systems Development Life Cycle (SDLC) methodology. The authors of this thesis re-evaluated the project's initial state and identified the following issues: the current website's design is out of date and should be updated; its implementation lacks a database, making data collection more difficult; and the new application's content should be maintainable without professional IT skills. To address these hurdles, the writers of this thesis created a web application using a headless CMS (WordPress) and React. The web application was completed to a working pilot level that could be utilised while additional development was carried out. Additionally, administrative workers received training in the form of video for the application developed.

In [12], the researcher is exploring the feasibility of building a CMS framework for public health professionals working in the Telehealth Department of the Malaysian Ministry of Health (MOH). The researcher expressed concern that the measures necessary by public health experts to locate information, data, specifications, materials, and clinical information will necessitate an exhaustive search of databases and archives. Due to the fact that this information is not contained on a single platform, responsible health professionals must have separate logins in

order to conduct information searches. By utilising a content management system (CMS), users ranging from clerks to public health specialists would be able to apply content from various sorts of documents and materials produced. Additionally, with suitable authorization rules in place in the majority of CMS, documents created or kept in the CMS will have access levels that restrict confidential or sensitive information to certain users inside the business.

According to research published in [28], headless CMS are preferred when editors have a technical background and the producing team prefers an agnostic approach to CMS implementation, whereas a traditional CMS with WYSIWYG features is preferred when stability and editorial independence are valued. The study presents a collection of factors that, depending on the environment in which the CMS is utilised and deployed, can be seen as benefits or drawbacks. The study focused on CMS because they saw it as a way to isolate content and code from the same tools as more web applications in the market grows in size and complexity.

The study in [9] delves into the concept of building your own Backend-as-a-Service (BaaS) platform using vendor-neutral techniques. The proposal's objective was to conceive, architect, and create a platform for heterogeneous micro-applications. According to the author, the BaaS platform is a crucial business resource that must be developed independently in order to avoid vendor lock-in, platform shutdown, or other detrimental changes. This is because many BaaS platforms company have failed and been shut down. These apps were developed in a monolithic fashion and were unable to evolve over time. The author concludes that to remain viable, the BaaS platform must grow to keep pace with the ever changing landscape of information technology (IT) and be heterogeneous in nature. Additionally, the paper discusses how to implement the platform.

Alacrity, a hybrid mobile application developed by Red Nuclear Monkey using the Flutter framework, is presented in the thesis in [4]. The work demonstrates Red Nuclear Monkey's technique for implementing multiple options of BaaS application within a Flutter project. Moreover, the thesis conducts a research of academic and industry literature to ascertain why businesses choose to utilise a Backend-as-a-Service solution versus a custom backend. Additionally, the study examines multiple BaaS, including Firebase, AWS Amplify, Parse, Azure Mobile Apps, and Kinvey. The study's scope was later narrowed to include simply a comparison of Firebase and Parse for usage in the Alacrity project. Finally, the analysis proposes that the company adopt Firebase, which looks to be the most accessible option that fits all of the recommendations of the company.

Firebase is also used as BaaS in another study at [29]. The author created an Android application for Mulund College of Commerce in the paper. It makes use of Firebase's features to demonstrate its utility as one of the finest mobile backend-as-a-service providers. The author made comparison with 3 other BaaS which are Amazon AWS Mobile, Parse and Back4App. Firebase proved to be an ideal fit for the project because it fits all of the project's basic requirements and also supports real-time changes as required by the application.

Results and Discussion:

Strapi

Strapi is an open-source headless CMS that allows developers to use their preferred tools and frameworks on the front-end while allowing editors to manage and distribute their content via their application's administration panel. Strapi is a content management system that is built on a plugin architecture. Users can make third-party plugins for the Wordpress CMS, which is initially beneficial because it enables the public to extend the CMS to accommodate a variety of use cases. However, over time, there is a great deal of complexity associated with it, primarily because plugins go unmaintained, resulting in defects and issues for plugin users. Rather than following the Wordpress model for plugin development, Strapi plugins are created by the core Strapi team if they believe the extensibility is required for the Strapi core, and are available for public contribution. Its admin panel and API are extendable, and each component is fully adaptable to fit any use case. Strapi also includes an integrated user management system that enables administrators and end users to manage their access rights in detail.

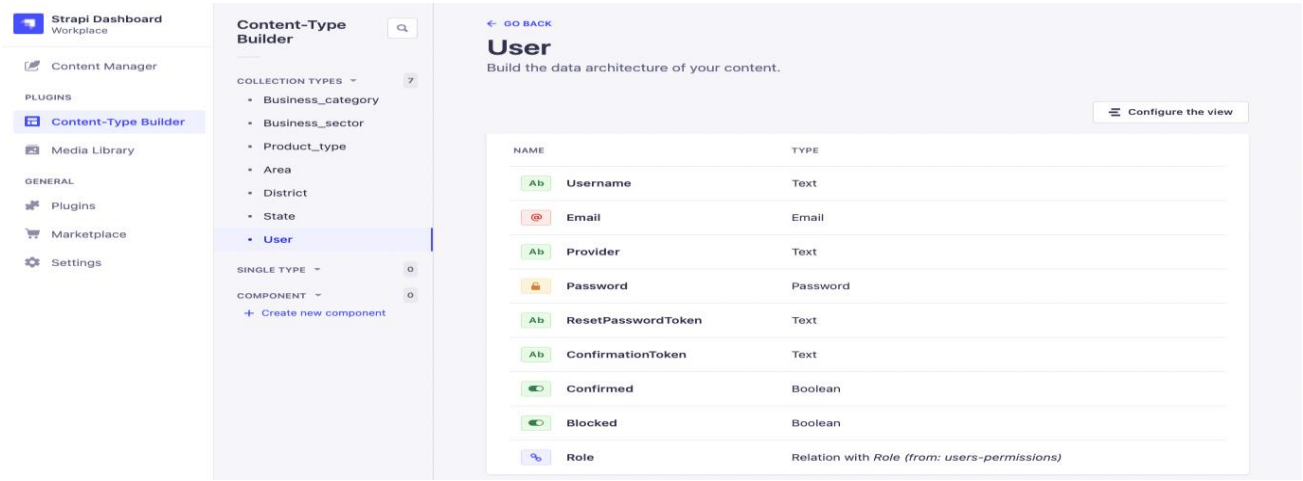

Figure 1. Content-Type Builder page in Strapi admin dashboard

Strapi enables users to construct and specify their data models quickly. By providing an admin graphical user interface, the Strapi system's administrator user will be able to establish database tables and the necessary relations to connect them. Additionally, Strapi assists in the generation of GraphQL and Restful API endpoints from the provided data model. This significantly reduces the amount of time developers spend developing the endpoints that will be used by frontend Javascript code. The development team's first-party support for plugins is also extremely beneficial; some of the most important expandable plugins built are those that handle authentication and authorisation. This plugin manages user registration, sign-in and sign-out, as well as access and refresh token management between the system and the front-end client. Apart from that, there are plugins that provide email handling for the system, as many systems rely on email as a proper notification route. It integrates with a variety of popular email services, including Amazon SES, Mailgun, and Sendgrid. Additionally, there is a plugin that assists with internationalisation management. This makes working with several languages

easier compared to many other CMS and is extremely beneficial for marketing purposes if the product is intended to be utilised by consumers from a range of different countries and areas.

Although Strapi has a plethora of advantages, it does have some disadvantages. To begin, while the users and permissions plugin is incredible, it is also quite limited, as the system can only be configured to employ a maximum of three custom roles for authorization purposes. If more customised roles are required, they are included in the paid category. Additionally, the database migration assistance is unsatisfactory, particularly in a production setting after using the system's 'build' mode. As illustrated in Figure 2, the 'autoReload' capability is not available by default when in 'build' mode; it is only available when in 'development' mode. Finally, the main code of Strapi is written in Javascript; this may cause some concern for developers who wish to inspect the code in order to tweak or contribute, as the current standard appears to strongly favour Typescript. Additionally, a recent study indicates that utilising Typescript effectively reduces the likelihood of a problem occurring by 15% [30].

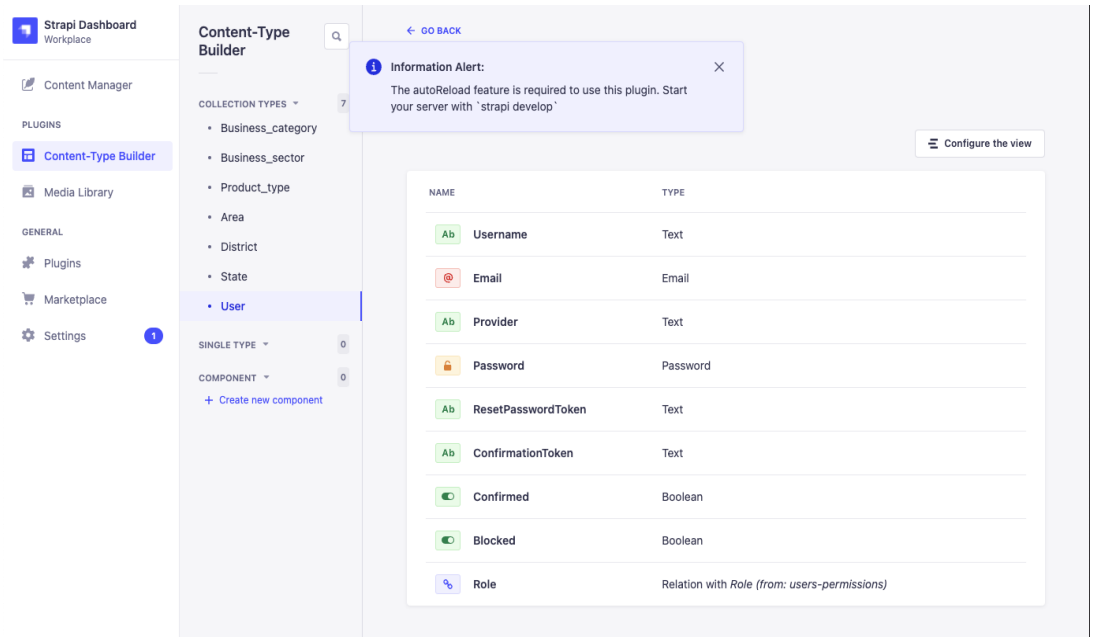

Figure 2. AutoReload feature is disabled in admin panel for “build” mode of Strapi by default.

Directus

Directus is a headless content management system (CMS) comparable to Strapi. They are open source, which means that anybody can read and modify the software, as well as contribute to the project. Additionally, they provide a Cloud pricing tier that lets users who do not wish to manage server deployment and management to delegate such responsibilities to the Directus core team. Directus users have the ability to create an infinite number of custom roles by default and does not have a price scheme for increasing the number of custom roles, as Strapi does. Directus's admin panel's graphical user interface, as well as its functionality on the content collection and admin settings pages, arguably look more modern and streamlined than Strapi's.

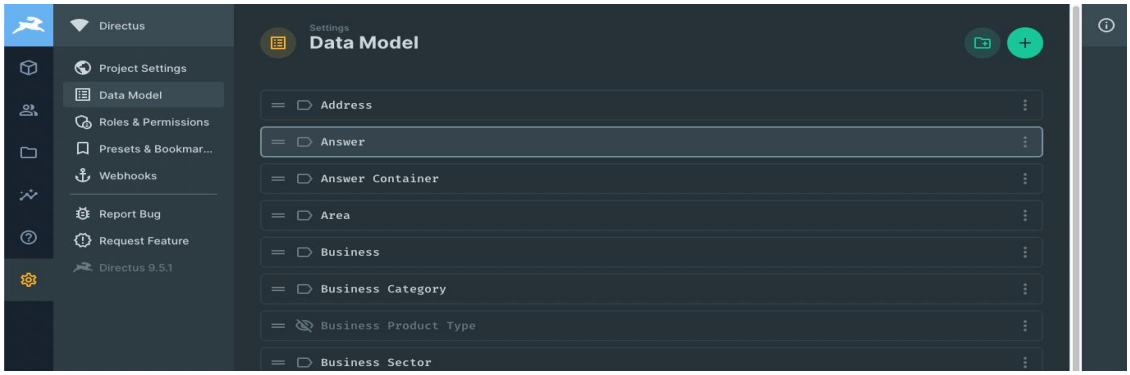

Figure 3. Data model settings page in Directus admin dashboard.

Directus has a number of advantages. To begin, Directus is extremely modular and flexible in its data architecture. It enables the user to use Directus with their own configured database structure. Directus will build the system data types required to hold its associated information, and when the user wishes to decouple Directus's logic from the system, the resulting Directus data type tables can be plug-off. This preserves the original defined database schema and does not rely on the one generated by Directus. Second, Directus' database migration scenario is far better to that of numerous other CMS. There are no restrictions on the generation of SQL schemas, even in the 'build' mode, and they also provide support for database schema migration if the user has a separate duplicate system. It is capable of identifying differences across schemas and implementing appropriate adjustments between systems. Furthermore, Directus builds GraphQL and Restful API endpoints in accordance with the defined database schema. Not only that, it has a Javascript SDK that front-end developers may utilise to wrap API calls and call them directly from Javascript functions. Finally, the Directus core code is written in Typescript for the back-end, and the front-end is built using the Vue.js framework.

The primary downside of Directus is that it is not as widely supported as some other more popular content management systems on the market, such as Strapi. As at the time of writing, Strapi has three times the number of Github stars as Directus. This directly correlates to the project's popularity. This also conveys a message to stakeholders when they choose it to be a part of their company's technology stack, as a more popular framework, library, or project implies a larger community of support.

Supabase

Supabase is a well-known open source alternative to Google's Firebase backend as a service (BaaS). Rather than relying on a NoSQL database, Supabase makes use of the open source and well maintained relational database Postgres. Supabase is available as a self-hosted solution or as a managed solution maintained by the Supabase team. As a BaaS application system, Supabase provides a number of core functionalities along with attractive features such as support for real-time data, integration with multiple popular authentication services, and email services that are required to build and bootstrap the backend functionality and simplify the process for front-end and mobile developers to use them.

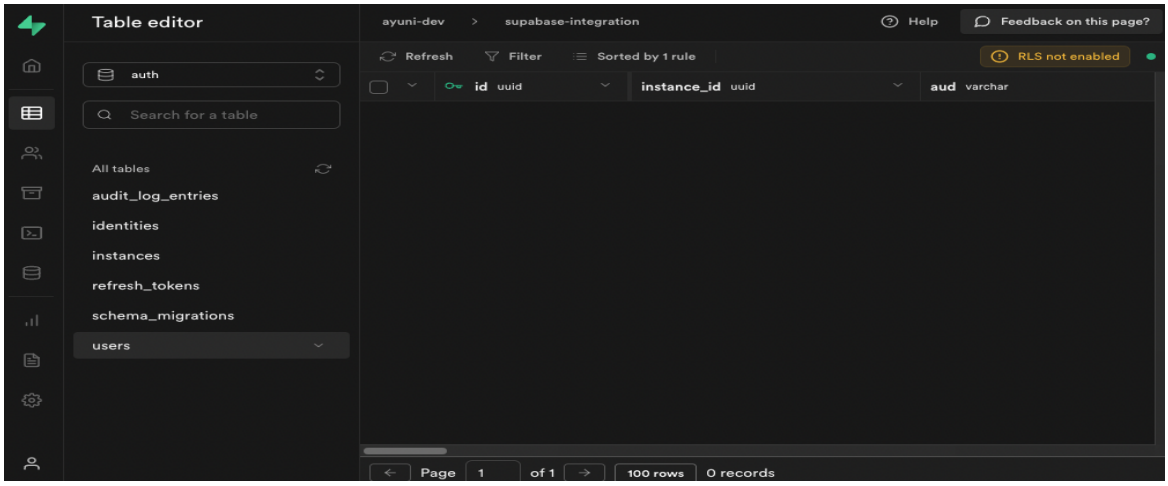

Figure 4. Table editor page in Supabase admin user interface dashboard.

Supabase features a plethora of advantages. To begin, the data architecture is quite flexible; it enables the user to choose their own Postgres database schema. Second, it supports real-time data transmission to clients via websocket. This functionality is extremely advantageous for projects that require the vital assistance of transmitting real-time data to clients. Thirdly, it assists in the generation of Restful API endpoints and their associated customised documentation. Additionally, it includes a Javascript SDK for front-end developers and a Dart SDK for mobile developers working with the Flutter framework.

In comparison to other popular BaaS like Firebase, Supabase comes with fewer out-of-the-box integration support; nonetheless, projects that require highly relational data will benefit from Supabase's datastore selection.

Table 1 summarize the comparison made between 2 headless CMS (Strapi and Directus) with a BaaS application (Supabase).

Table 1. Summarize comparison between Strapi, Directus and Supabase.

Features	Strapi	Directus	Supabase
Modularity of Data Schema	No	Yes	Yes
Authentication (register, login, logout)	Yes	Yes	Yes
Out of the box support for multiple authentication providers	Yes (plugin)	No	Yes
Good documentation	Yes	Yes	Yes
Example starter-kits integration available	Yes	No	Yes
Authorization (Unlimited custom roles)	No	Yes	Yes
Non-technical user and content writer friendly	Yes	Yes	No
GraphQL generated based on schema	Yes	Yes	Yes (pg_graphql)
Github popularity	1	3	2

Conclusion:

In this study, we compare two headless CMSs, Strapi and Directus, and one BaaS application, Supabase, which will serve as the backend for the e-Suripreneur system. Directus appears to be the most suited solution for our system based on the comparison results, as it fits all of the requirements of the e-Suripreneur project. In future work, we will investigate the optimal option for integrating the Directus system chosen for the e-Suripreneur backend with a third-party service to meet a more sophisticated requirement of the system.

Acknowledgement

The authors would like to thank the Ministry of Higher Education, Malaysia for financial support through the Long-term Research Grant Scheme (LRGS) 600-RMC/LRGS 5/3 (001/2020).

References

1. G. X. Carmine, Wang Pekka, Abrahamsson, "Agile Processes in Software Engineering and Extreme Programming," International Conference on Agile Software Development, pp. 52-63, 2015, doi: 10.1007/978-3-319-18612-2.
2. G. X. Carmine, Wang Pekka, Abrahamsson, "Why Early-Stage Software Startups Fail: A Behavioral Framework," International Conference of Software Business, pp. 27-41, 2014.
3. G. M. Carmine, Unterkalmsteiner Nicolò, Paternoster Tony, Gorschek Pekka, Abrahamsson, "What Do We Know about Software Development in Startups?," IEEE Software, vol. 31, no. 5, pp. 28-32, 2014.
4. H. Allain, "Improving productivity and reducing costs of mobile app development with Flutter and Backend-as-a-Service," Master's Programme in ICT Innovation, Aalto University, 2020.
5. M. Dudjak and G. Martinović, "An API-first methodology for designing a microservice-based Backend as a Service platform," Information Technology And Control, vol. 49, no. 2, pp. 206-223, 2020, doi: 10.5755/j01.itc.49.2.23757.
6. R. S. Siegfried, Arzt Robert, Hahn Max, Kolhagen Eric, Bodden, "(In)Security of Backend-as-a-Service," 2016.
7. W. Wingerath, F. Gessert, E. Witt, S. Friedrich, and N. Ritter, "Real-Time Data Management for Big Data," in EDBT, 2018, pp. 524-527.
8. F. Gropengießer and K.-U. Sattler, "Database backend as a service: automatic generation, deployment, and management of database backends for mobile applications," Datenbank-Spektrum, vol. 14, no. 2, pp. 85-95, 2014.
9. B. Carter, "Grow your own Backend-as-a-Service (BaaS) platform," presented at the GOCICT 2015 Conference College of Information & Computer Technology, 2015.
10. Deepak Mathur, N. K. V. . (2022). Analysis & Prediction of Road Accident Data for NH-19/44. International Journal on Recent Technologies in Mechanical and Electrical Engineering, 9(2), 13–33. https://doi.org/10.17762/ijrmee.v9i2.366
11. Y. Tjong, "Successful measurement of Content Management System implementation," presented at the International Conference on Information Management and Technology (ICIMTech), 2016.

12. K. B. Marin, Vukelić Tamara, Rojko, "CONTENT MANAGEMENT SYSTEM SECURITY," Zbornik Veleučilišta u Rijeci, vol. 4, pp. 29-44, 2016.
13. H. Hashim, "Content Management System (CMS) for Public Health Professional in theTelehealth Department, Ministry of Health, Malaysia: A Conceptual Framework," International Journal of Innovation, Management and Technology, vol. 6, no. 1, 2015, doi: 10.7763/ijimt.2015.V6.574.
14. A. R. Trikusuma, N. Q. Nada, and M. Novita, "The Web-based Application of Small and Medium Enterprises (SMEs) Product Distribution Management with Content Management System Shopify Integration in Netasia Singapore," Advance Sustainable Science, Engineering and Technology (ASSET), vol. 4, no. 1, p. 0220105, 2022.
15. K. S. Yoon and Y. H. Kim, "A Design and Implementation of Integrated Content Management System Based on Microservices Architecture," KIPS Transactions on Software and Data Engineering, vol. 8, no. 3, pp. 97-108, 2019.
16. Malla, S., M. J. . Meena, O. . Reddy. R, V. . Mahalakshmi, and A. . Balobaid. “A Study on Fish Classification Techniques Using Convolutional Neural Networks on Highly Challenged Underwater Images”. International Journal on Recent and Innovation Trends in Computing and Communication, vol. 10, no. 4, Apr. 2022, pp. 01-09, doi:10.17762/ijritcc.v10i4.5524.
17. A. Ganapathy, "Cascading Cache Layer in Content Management System," Asian Business Review, vol. 8, no. 3, pp. # 24-182, 2018.
18. R. R. A. Hussein and A. B. Al-Kaddo, "E-Learning by using content management system (CMS)," International Journal of Advanced Computer Science and Applications, vol. 5, no. 10, 2014.
19. B. F. RD, Al-Mahri R, Malathi, "A Perspective Study on Content Management in E-Learning and M-Learning," 2016.
20. R. S. Sandeep, HerringAllison, Gray, "Identifying an appropriate Content Management System to develop Clinical Practice Guidelines: A perspective," 2015, doi: https://doi.org/10.1177/1460458215616264.
21. V. Ghorecha and C. Bhatt, "A guide for selecting content management system for web application development," International Journal, vol. 1, no. 3, pp. 13-17, 2013.
22. D. Priefer, P. Kneisel, and G. Taentzer, "JooMDD: A model-driven development environment for web content management system extensions," in 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C), 2016: IEEE, pp. 633-636.
23. S. Wan, D. Li, and J. Gao, "Exploring the Advantages of Content Management Systems for Managing Engineering Knowledge in Product-service Systems," Procedia CIRP, vol. 56, pp. 446-450, 2016, doi: 10.1016/j.procir.2016.10.087.
24. Philip, A. M., and D. S. . Hemalatha. “Identifying Arrhythmias Based on ECG Classification Using Enhanced-PCA and Enhanced-SVM Methods”. International Journal on Recent and Innovation Trends in Computing and Communication, vol. 10, no. 5, May 2022, pp. 01-12, doi:10.17762/ijritcc.v10i5.5542.
25. S. Sivankalai, "Academic Libraries' Content Management Trends," Library Philosophy and Practice, vol. 8, pp. 1-13, 2021.

26. M. Tanner, "Implementation of the new Swiss-Hema website using headless CMS and React.js," Business Information Technology, Haaga-Helia University of Applied Sciences, 2020.
27. "Strapi - Open source Node.js Headless CMS." https://strapi.io/ (accessed 5/2/2022, 2022).
28. "Directus: Open Data Platform for Headless Content Management." https://directus.io/ (accessed 5/2/2022, 2022).
29. "The Open Source Firebase Alternative | Supabase." https://supabase.com/ (accessed 5/2/2022, 2022).
30. Ghazaly, N. M. . (2022). Data Catalogue Approaches, Implementation and Adoption: A Study of Purpose of Data Catalogue. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(1), 01–04. https://doi.org/10.17762/ijfrcsce.v8i1.2063
31. L. P. Pyykölä, Edina, "Design and development of a web application for Vihdin-Nummelan Kylähistoria Ry with WordPress as a Headless CMS and React," Business Information Technology, Laurea University of Applied Sciences, 2020.
32. L. Öfverstedt, "Why go headless – a comperative study between traditional CMS and the emerging headless trend," Uppsala universitet, 2018.
33. H. Dand and D. Sharma, "Firebase as BaaS for College Android Application," International Journal of Computer Applications, vol. 178, no. 20, pp. 1-6, 2019, doi: 10.5120/ijca2019918977.
34. Z. Gao, C. Bird, and E. T. Barr, "To Type or Not to Type: Quantifying Detectable Bugs in JavaScript," presented at the 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), 2017.
35. N. A. Farooqui, A. K. Mishra, and R. Mehra, “IOT based Automated Greenhouse Using Machine Learning Approach”, Int J Intell Syst Appl Eng, vol. 10, no. 2, pp. 226–231, May 2022.
36. Nakhaei A, Soltani F. Modelling and Optimisation in the Design of Pipeline Network Systems Using Ant Colony Optimisation Algorithm (ACO). sjis. 2021; 3 (4) :1-17, URL: http://sjis.srpub.org/article-5-131-en.html