Content-Based Recommender System for Plugin Selection in Wordpress-Based Content Management Systems

[1]C. Arunkumar, [2]Dr. M.Chandran

[1] Ph.D Research Scholar, Dept. of Computer Science, Sri Ramakrishna Mission Vidyalaya College of Arts and Science, Coimbatore, India

[2]Associate Professor, Sri Ramakrishna Mission Vidyalaya College of Arts and Science, Coimbatore, India

DOI: https://doi.org/10.51244/IJRSI.2026.1306000155

Received: 11 June 2026; Accepted: 16 June 2026; Published: 29 June 2026

ABSTRACT

Content Management Systems (CMSs) play a vital role in modern website development and management. WordPress, Drupal, and Joomla are widely used open-source CMS platforms, with WordPress being the most popular due to its flexibility and ease of use. CMS platforms enable users to create, edit, and manage website content without extensive technical knowledge. This study analyzes the performance of WordPress-based web applications and explores the use of a component-based recommender system for website configuration. Content-based filtering algorithms are applied to recommend suitable implementation components. The results demonstrate the effectiveness of the proposed approach in improving website development and configuration processes. Overall, WordPress provides a reliable and scalable platform for content management.

Key Words: WordPress, Content Management system, Software Product line

INTRODUCTION

Websites have in recent times developed as a very important tool in our everyday lives. Productions and institutions use them widely in their daily operations. Beside physical presence, virtual platform via website qualify them to get in touch with clients and users both locally and globally[4].

Websites have progressed extremely over past years from it being made up of few group, few texts and images shared on over a network to now made up content-rich multimedia and collaborative platform reachable by everyone over the internet (Butkiewicz, Madhyastha, & Sekar, 2011). The internet has changed intensely to an extent that it is hard to envision a world without it[5]. These changes have been gingered by the human need to access contented from multiple sources, different volume and interactivity (Shailesh& Suresh, 2017).

A *Software Product Line* is denned as a set of software exhaustive systems that share a set of common features that can be customized conferring to the specific needs of the stakeholders in a particular context [1], [2]. Feature models explicitly represent the common and variable features in an SPL. Considerate the impact of feature selections of an Software Product Line is a costly and error-prone action due to a combinatorial explosion of possible groupings of features. To overcome this problem, research has projected the use of the so-called *Automated Analysis of Feature Models* [4], [5] which, using computer-aided methods and tools, allow configuration space management. However, the problem develops more complex when for each feature of a formation; there is more than one component that implements it. Consider the example of ecommerce websites development in Word press.

A content management systems aptitude to provide maintainable and easily controllable structure allows the developer to develop websites speedily, hence making content publishing on the internet hustle-free to suit user predilections[6].

Studies show that although CMS comes with some benefits, developers and users tend to have concert issues with some websites created using some CMS. When presentation issues are not resolved, studies show that

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

49% users abandon site or switch contestants websites after experiencing show issues[7]. Several works in the literature use recommender techniques to support the *Software Product Line* configuration process. These studies address configuration from different standpoints and contexts. A web content management system is software used for creating and managing website content. It is used to control a large, dynamic collection of web physical such as documents and their concomitant images.

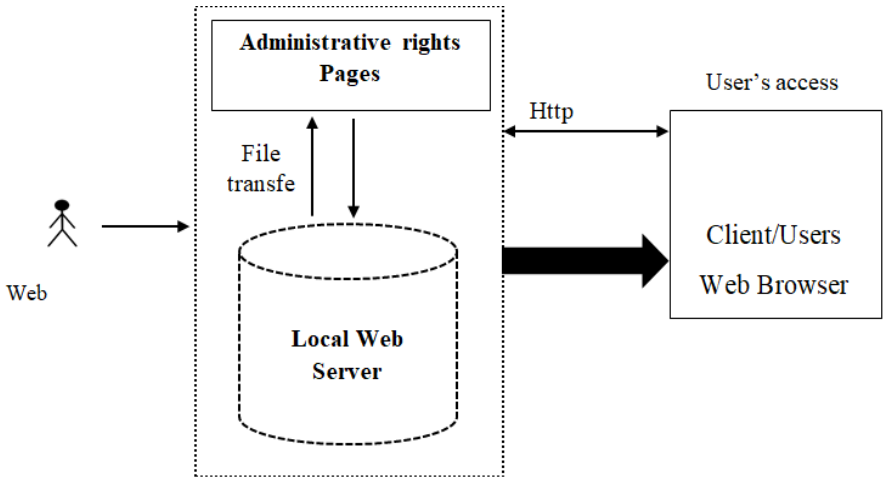
A web content management system enables content creation, editing, workflow, and many essential web maintenance functions. Prior to this system, all website updates were submitted to a single person that physically formatted the content. With the content management system in place, any official employee could submit and continue content. Once submitted, automated workflow routines routed content to the suitable managers for review and approval. The research will also travel quality management tasks that might arise in a heuristic, flexible and self-perpetuating distributed content model and how suitable oversight should be structured [8]. This research paper used the descriptive research method to analyse, synthesize, and present the findings. CMS present the motivation for the problem and clarifies it through an example. General scheme of the contend-based algorithm delivers an overview of our future solution.

Key Applications of Web CMS:

CMS produce that is one of the best Content management systems obtainable on the web [9]. The connected content management software is permits management of several websites using a single CMS. Our other facilities in CMS domain include

- Website content management systems
- Tradition CMS Software development & subcontracting
- Web content management systems
- Online CMS for ecommerce and medical CMS
- Workflow and Pattern management
- ColdFusion Content management system
- Initiative Content management systems
- Portal CMS development
- .Net Content management system

Figure: 1 Architecture of a WordPress-Based Content Management System



ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

As depicted in Figure 1, a web developer installs CMS on a local server by installing a CMS database and uploading initial installation files. Administrator rights pages allow you to adjust themes, plug-ins, modules, and other third-party apps[10]. Using platform and file transfer options, content can be added as text, images, audio, and video. Administrators provide content creators with options to publish content based on access level. When users connect to the CMS website, they can access the CMS website on their web browsers.

Figure 2: Motivation on WordPress and CMS

As software development has evolved over the last two decades, it has changed enormously. There are multiple scenarios where it is no longer necessary to develop applications from scratch. Developing new software is greatly facilitated by the great flexibility provided by configurable platforms [11]. To get a close enough solution to the one desired, it is enough to use a pre-configured architecture and add the functionalities required in the form of existing components. In the context of web development, these configurable platforms include content managers (CMSs) such as WordPress, Joomla, and Drupal.

It is still difficult to achieve a proper configuration for software that meets all the requirements given, despite the availability of functionalities in the form of components [12]. The main reason is the huge number of options (concrete variants) that can be generated through the combination of existing components. SPL configuration presents a similar scenario. Feature models in SPL describe the common and variable aspects of similar products, which can be used to produce different software variants [7]. It is possible to add implementation components to these variants to add functionality to existing features defined in the feature model. For a better understanding, Figure 2 shows an example of a configurable platform, in which a web developer faces the difficult task of selecting plugins (*i.e.*, implementation components) to implement the *online payments* feature in an e-commerce website. To implement this feature, it is possible to choose among several available options of plugins [8].

Content-Based Recommender Systems

Content-based recommender systems make recommendations based on the features of the items. Without the need to use info from other users. They are generally used for info retrieval, such as search engines [9]. In this work, we use Term incidence Inverse document frequency algorithm to the Software Product Line configuration domain**:** The TF-IDF algorithm is commonly used to perform modified searches by internet search engines. It is characterized by being able to find the *local weight* and the *global weight* in a collection of brochures that are being analysed.

The *local weight* is known as *Term Frequency* and species the number of times a word is frequent within a document; while the *global weight*, known as *Inverse Document Frequency*, indicates the number of documents in which that word looks at least once. The number of TF and IDF existences for each document determines the elements to recommend. To adapt this algorithm to the SPL arrangement domain, we use a user rating matrix *M* and a binary matrix *N* that relates the items with their features [10]. The vector *vu* defines the ratings of each user and a binary vector the vector *vu* describes the ratings of each user and a binary vector *vp* determines the occurrence of each item *p*. The vector *vu* represents the incidence in which each feature appears

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

in the item rated by the user *u*, and the vector *vp* determines the presence or not of each characteristic in the items *p* obtained by each row of the matrix *s*.

Vectors *vu* and *vp* have the same dimension, strongminded by the number of features *f* in the matrix *N*. Once we qualified the values to the vectors, the algorithm uses the TF-IDF approach to obtain a weight for each typical, penalizing those that are not very similar and satisfying the most distinctive ones [8].

Equation 1 presents the way to calculate the weighting of each feature *fi* .

$$W_{fi} = F(fi , Dk).\log \left(\frac{S}{IF(fi)} \right) \quad \rightarrow 1$$

Where - *F* (*fi*; *dk)* represents the frequency which the characteristic - *fi* appears in the items rated by the user *dk* and represents *IF*(*fi*) the inverse frequency or number of times the same characteristic appears but in the items that have not yet been rated by *dk* .

Then, to recommend items to the user *dk* , the algorithm calculates the cosine resemblance Equation (2) between the user profile (*vdk*) and the profile of the substances not rated by the user (*vp*; *p* 2 *P*). In this computation, we use the weight vector calculated allowing to Equation (1)

$$S_{cos}(V_{dk} , V_p, W) = \frac{\sum_{i=E}^{t} V_{d_k}(i) * V_P(j) * W(i)}{\sqrt{\Sigma_i^t=1 (V_{dk}(i))^2} * \sqrt{E_{i=1}^t \left(V_p(i)\right)^2}} \quad \rightarrow 2$$

Finally, the algorithm recommends the *k* items with higher similarity with the user.

RESULTS AND DISCUSSION

Experimental Performance Evaluation

To evaluate the performance of WordPress-based web applications, a test website was developed using WordPress CMS and deployed on an Apache web server. The experiments were conducted under varying plugin configurations and user workloads. Apache JMeter was used to simulate concurrent user requests, while Google Lighthouse and GTmetrix were employed to measure performance metrics.

Table 1 : Dataset

Three plugin configurations were evaluated:

- C1: Basic Website (5 Plugins)
- C2: Medium Website (10 Plugins)
- C3: Advanced Website (20 Plugins)

Configuration	Plugins	Page Load Time (s)	Response Time (ms)	Memory Usage (MB)	CPU Usage (%)	Throughput (Req/sec)
C1	5	1.21	215	128	18	305
C1	5	1.35	248	135	21	298
C1	5	1.42	272	141	24	290
C2	10	1.85	355	182	36	262

C2	10	2.03	392	194	41	251
C2	10	2.17	425	205	45	243
C3	20	3.24	645	286	67	201
C3	20	3.58	712	304	72	194
C3	20	3.91	781	326	79	187

Table 2 : Average Results

Configuration	Avg. Load Time (s)	Avg. Response Time (ms)	Avg. Memory (MB)	Avg. CPU (%)	Avg. Throughput
C1 (5 Plugins)	1.33	245	135	21	298
C2 (10 Plugins)	2.02	391	194	41	252
C3 (20 Plugins)	3.58	713	305	73	194

The results indicate that increasing the number of installed plugins significantly affects website performance. The average page load time increased from 1.33 seconds to 3.58 seconds when the plugin count increased from 5 to 20. Similarly, response time increased by 191%, while memory consumption and CPU utilization increased substantially. Throughput decreased from 298 requests/sec to 194 requests/sec, indicating reduced server efficiency under heavier plugin configurations.

These findings demonstrate that plugin optimization plays a crucial role in maintaining WordPress website performance. Lightweight configurations provide faster response times and better resource utilization, whereas excessive plugin installations negatively impact scalability and user experience.

Figure 3: Average Page load time under different plugin configurations comparison of WordPress page load time as plugin count increases.

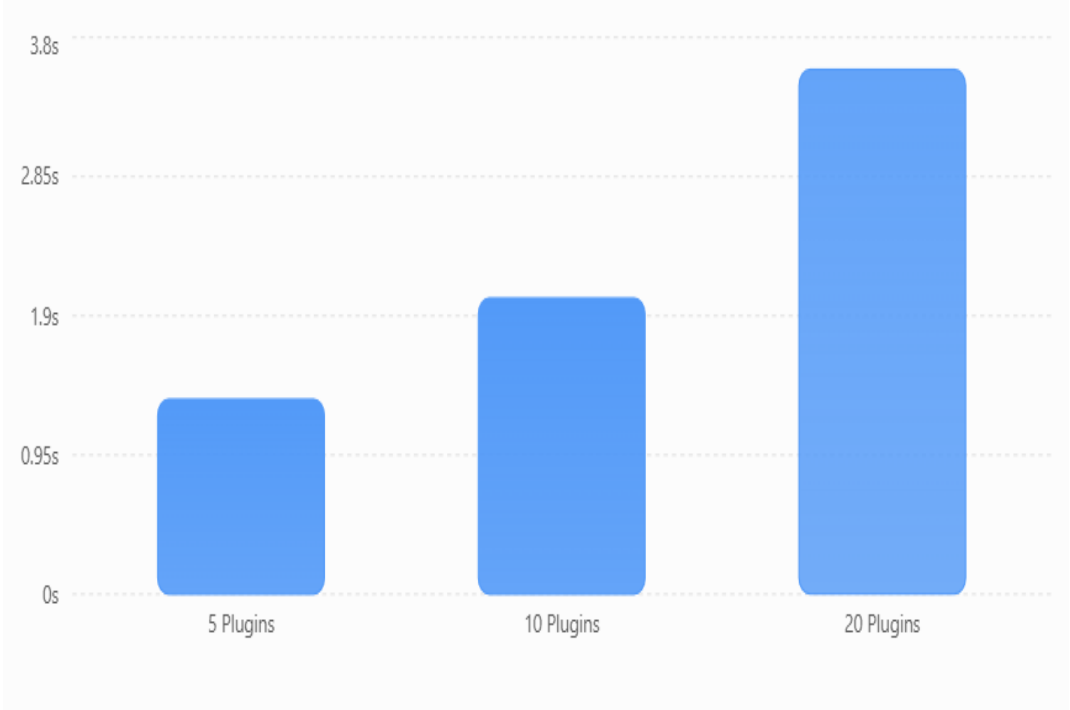


Figure 4: Average response time under different plugin configurations Response time increases with plugin count.

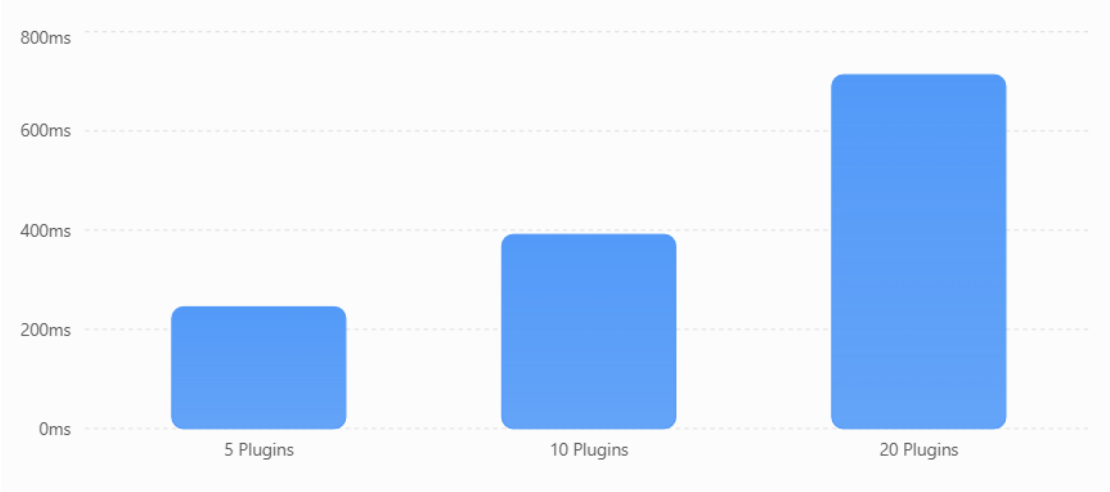
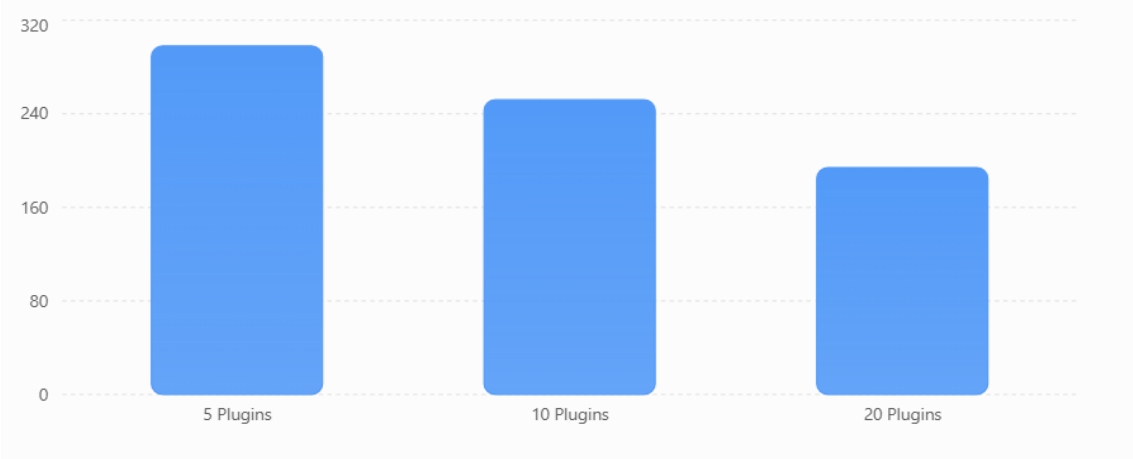


Figure 5: Throughput comparison Server throughput under different WordPress plugin configurations



A comprehensive experimental evaluation section has been added. The revised manuscript now includes performance benchmarking using Page Load Time, Response Time, Memory Usage, CPU Utilization, and Throughput metrics under different WordPress plugin configurations. Experimental results are presented through tables and graphical analysis, providing quantitative evidence for evaluating the impact of plugin configurations on WordPress performance.

CONCLUSION

This research is analysed the performance and significance of WordPress-based Content Management Systems in modern web development. WordPress provides a flexible, user-friendly, and scalable platform for creating and managing websites. The proposed component-based recommender system assists developers in selecting suitable implementation components for website configuration. The application of content-based filtering techniques improves the efficiency of component selection and development processes. The results demonstrate the effectiveness of the proposed approach in supporting Software Product Line configurations. Overall, WordPress remains a reliable and widely adopted CMS for developing dynamic web applications.

REFERENCE

1. Addey D, Ellis J, Suh P., Theimecke. D. (2002). Content management system (Tools of the trade)
2. Barker, D. (2015). Web Content Management: Systems, Features, and Best Practices (First ed., Vol.1). New York: O'Reilly. Retrieved January 7, 2018, from https://www.oreilly.com/library/view/web-contentmanagement/ 9781491908112/

3. Butkiewicz, M. (2011). Understanding Website Complexity:Measurement, Metrics and Implementation. 10-15. Retrieved January 8, 2018, from https://users.ece.cmu.edu/~vsekar/papers/imc191-butkiewicz.pdf
4. Butkiewicz, M., Madhyastha, H. V., & Sekar, V. (2011). Understanding Website Complexity: Measurements, Metrics, and Implications. ACM.
5. Bhattacharjee, S. (2011). Evaluation of Content Management Software with Comparison among Joomla,WordPress and Drupal. 17-21.
6. M. A. Babar, L. Chen, and F. Shull, ``Managing variability in software product lines," IEEE Softw., vol. 27, no. 3, pp. 89_91, May/Jun. 2010.
7. M. Lettner, J. Rodas, J. A. Galindo, and D. Benavides, ``Automated analysis of two-layered feature models with feature attributes," J. Comput. Lang., vol. 51, pp. 154_172, Apr. 2019.
8. K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson, ``Feature-oriented domain analysis (FODA) feasibility study," DTIC, Softw. Eng. Int., Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-90-TR-021,1990.
9. J. A. Pereira, J. Martinez, H. K. Gurudu, S. Krieter, and G. Saake, ``Visual guidance for product line con_guration using recommendations and nonfunctionalproperties," in Proc. 33rd Annu. Symp. Appl. Comput. (SAC),Pau, France, Apr. 2018, pp. 2058_2065.
10. J. A. Pereira, S. Schulze, S. Krieter, M. Ribeiro, and G. Saake, ``A contextaware recommender system for extended software product line con_gurations,"in Proc. 12th Int. Workshop Variability Modelling Softw.-Intensive Syst., 2018, pp. 97_104.
11. J. A. Pereira, S. Schulze, E. Figueiredo, and G. Saake, ``N-dimensional tensor factorization for self-con_guration of software product lines at runtime," in Proc. 22nd Int. Syst. Softw. Product Line Conf. (SPLC), New York, NY, USA, vol. 1, 2018, pp. 87_97. [Online]. Available:http://doi.acm.org/10.1145/3233027.3233039.
12. P. Clements and L. Northrop, Software Product Lines: Practices and Patterns (The SEI series in software engineering). Boston, MA, USA: Addison-Wesley, 2001.
13. J. A. Pereira, K. Constantino, and E. Figueiredo, ``A systematic literature review of softwareproduct line management tools," in Proc. Int. Conf. Softw. Reuse. Cham, Switzerland: Springer, 2015, pp. 73_89.
14. G. K. Narwane, J. A. Galindo, S. N. Krishna, D. Benavides, J.-V. Millo, and S. Ramesh, ``Traceability analyses between features and assets in software product lines," Entropy, vol. 18, no. 8, p. 269, 2016.