INCITEST
International Conference on Informatics Engineering, Science & Technology
UNIKOM

Website-based Arabic Conjugation (*Sharf*) Learning*:* *ArabicMorph*

Moh Naufal Faqih, Firman Firdaus, Ryan Azis Saputra, Usep Mohamad Ishaq*, Hidayat Hidayat

Faculty of Engineering and Computer Science, Universitas Komputer Indonesia, Bandung, Indonesia

*Corresponding E-mail: usep.mohamad.ishaq@email.unikom.ac.id

Abstract. Learning Arabic morphology (*sharf)*, particularly verb conjugation, presents a significant challenge for students due to its complexity and reliance on rote memorization. This research aims to design and develop *ArabicMorph,* an interactive web application to facilitate the automatic and efficient learning of *sharf*. The system was developed using the Waterfall model, with requirements prioritized through the MoSCoW method. It is built on the Laravel framework and integrates the Qutrub API for verb conjugation and the DeepSeek AI API for contextual translation. System evaluation was conducted via beta testing with a questionnaire administered to 17 respondents to measure user satisfaction. The results show that all core features were successfully implemented, and the system achieved a very high user satisfaction level with an average score of 91.8%, covering functionality, appearance, and ease of use. These findings indicate that the integration of rule-based linguistic technology with artificial intelligence in a modern interface is effective in overcoming the difficulties of leaning *sharf*. In conclusion, *ArabicMorph* has been successfully realized as a practical, adaptive, and user-friendly tool for independent training. The impact of this research is a digital solution that can potentially enhance student interest and understanding of Arabic morphology, providing a foundation for future development with additional educational features.

Keywords: *ArabicMorph*, *sharf*, *tashrif*, Laravel, Qutrub, Deepseek, Arabic Language Learning, API.

1. Introduction

Arabic has a rich linguistic heritage, with morphology or *sharf* being one of its main pillars, which specifically studies the internal structure and formation of words. In the tradition of Arabic grammar, the science of *sharf* is often likened to the “mother of all sciences” (*Ummul 'Ulum*), while the science of *Nahwu* (syntax) is its “father” (Anwar, 2019). This expression underscores the fundamental role of *Sharf* as the foundation for understanding the original meaning and potential of each word before it is incorporated into a sentence, as it is impossible to comprehend the relationships between words in a sentence (*nahwu*) without first understanding the words themselves (*sharf*). This complexity is particularly evident in the process of verb conjugation (*Tashriful fi’li*), which is a central component of Arabic morphology due to its richness and complexity.

However, behind its fundamental position and theoretical richness, the complexity of *sharf* science raises a series of real problems faced by Arabic language learners today. The main obstacles in learning *sharf* include the large number of words change patterns that must be mastered and learning models that are often rote memorization. This memorization process is often perceived as a burden and a difficult aspect by some students (Yunisa, 2022).

As a result, many learners find it difficult to understand and use vocabulary according to the correct rules, which ultimately leads to low enthusiasm and the perception that *sharf* is a difficult subject to master.

Difficulties in learning *sharf* are also encountered at the Al-Hidayah Islamic Boarding School in Purwakarta. Through initial observations of the *sharf* learning process, many students experience difficulties in understanding and memorizing the patterns of word changes in Sharf. This has an impact on the slow development of Arabic language proficiency among students, especially in reading and understanding classical Arabic texts.

In response to this challenge, the *ArabicMorph* website was designed as a digital solution that integrates linguistic technology to simplify and visualize the conjugation process automatically. This system utilizes the Qutrub API, an open-source conjugation engine developed by Taha Zerrouki within a rule-based Arabic morphology processing platform, to automate the conjugation of verbs based on their roots and link them to various pronouns (Zerrouki, 2020). However, the novelty of this system lies not only in the use of the Qutrub engine but also in the redesign of the user experience through a more interactive and user-friendly Laravel interface, as well as the integration of AI-based translation services (DeepSeek) to provide direct translations of conjugated verb forms. Additionally, this system can also serve as a quick reference tool and reminder when learners forget the patterns of a word's changes, making it highly relevant for both self-learning and structured teaching.

This study aims to identify the main problems faced by *santri* and students in learning *sharf*, particularly in relation to difficulties in understanding verb conjugation patterns in Arabic. As a solution, a digital platform in the form of an interactive website has been developed to assist in the automatic learning process of *sharf*. The website is equipped with a search feature that can display complete verb conjugation patterns (*sharf*) along with their meanings and morphological information. With a modern and interactive technology-based approach, it is hoped that this platform will enhance users' interest in learning and their understanding of the Arabic language.

2. Literature Review

This research is inseparable from supporting theories for studying and designing systems that are expected to function optimally. The following are supporting theories that reinforce this paper:

2.1 Software Development Model

The Waterfall model is one of the classic software development models that is linear and sequential. This model was first introduced by Winston W. Royce in 1970 and has since become the basic model in software engineering. The Waterfall model divides the system development process into several main stages: (1) requirement analysis, (2) system and software design, (3) implementation, (4) testing, and (5) maintenance. Each stage must be fully completed before proceeding to the next stage (Pressman & Maxim, 2015).

This model is ideal when system requirements are clearly understood from the outset and the risk of change is relatively low. One of the main advantages of the Waterfall model is

the well-structured documentation at each phase, which makes it easier for developers and stakeholders to understand the development process as a whole (I. Sommerville, 2015). In the context of the *ArabicMorph* system development, the Waterfall model was chosen because its development process follows systematic stages: starting with user needs analysis (particularly for Arabic language learners), designing the morphological structure and root word database, implementing the system with integration of the Qutrub and DeepSeek APIs, and finally the testing and refinement process.

2.2 Laravel and Model-View-Controller (MVC) Architecture

Laravel is an open-source PHP framework designed to build elegant and efficient web applications using the Model-View-Controller (MVC) architectural pattern. The MVC architecture divides an application into three main components: Model (data logic and database interactions), View (user interface), and Controller (the bridge between the model and view, as well as the application flow controller) (Stauffer, 2023)..Meanwhile, the **View** component is responsible for displaying the results of verb conjugation and translations in the form of an interactive table that is easy for users to understand.

The advantages of Laravel lie not only in its clean and intuitive syntax, but also in its comprehensive official documentation and extensive global community support, making it one of the most popular PHP frameworks in the world of web development today

2.3 Relational Database Design (MySQL) and Integrating Web APIs into Laravel-based Application Systems

Relational databases are the most commonly used data management system approach in modern software development. This model stores data in the form of tables that are interconnected through relationships, where each table has attributes (columns) and entities (rows) that can be uniquely identified using a primary key. MySQL is one of the open-source relational database management systems (RDBMS) that has been widely used across various web platforms (Dubois, 2013). For the construction of ArabicMorph, the API serves as an intermediary between the Laravel system and two outsourced services: Qutrub API, which generates Arabic verb conjugation according to word roots and morphological patterns; and DeepSeek AI API that automatically provides translation for conjugated words. API integration One of the most crucial aspects of integrating an API is error handling and fallback solution – such as being able to still alert users without failure when the API does not respond. Such practices are part

2.4 Unified Modelling Language (UML)

UML is a standard modeling language and now become an international standard for system docuentation and design in software angineering used to build object-based software systems (Booch et al., 2017). UML provides a number of diagrams that can be used at various stages of system development. These diagrams are classified into two primary groups: The first is **Structural Diagrams** comprising Class Diagrams, Component Diagrams, and Deployment Diagrams, which describe the static structure of a system and the second group is **Behavioural Diagrams** including Use Case Diagrams, Activity Diagrams, and Sequence Diagrams, which describe the flow, interactions, and dynamic processes within a system.

3. Methods

The Waterfall model is applied in the development of the *ArabicMorph* system. The Waterfall model is a software life cycle process that occurs linearly and sequentially" (Sukamto, Rosa Ariani & Salahuddin, 2016). This method was specifically chosen for its high relevance to the project's characteristics, as its development process followed systematic and well-defined stages, from a clear understanding of needs at the outset to final testing. Each stage in this model must be fully completed before proceeding to the next, ensuring a structured and well-documented process. The Waterfall stages model are shown in **Figure 1** below

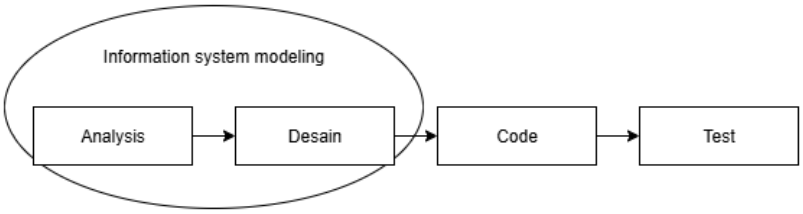

Figure 1. Waterfall Model Stages.

The process begins with the **Requirement Analysis** stage. In this stage, intensive requirements gathering was conducted through field research, literature studies, interviews, and documentation to detail the software specifications for a deep user understanding. Once all requirements were identified and approved, the research proceeded to the **Design** stage. This multi-step process on creating the system's blueprint, including its software architecture, data structures, interface representation, and coding procedures. Subsequently, the result of the design was realized in the **Implementation** (Code) stage, where the system design was translated into machine-executable code. After the software was built, it entered the **Testing** phase. This stage focused on the software's logic and functionality to ensure all parts were thoroughly tested. The main goal was to minimize errors and guarantee that the resulting output was as desired. The final stage of the model is **Maintenance**, which covers adjustments or feature additions after the software is in use, a process that can repeat the development cycle from the beginning for any changes to the existing software.

4. Results and Discussion

4.1 Requirement Analysis

During the Requirement Analysis stage of the Waterfall software development model, observations were made on target user characteristics, and a literature review was conducted on *Sharf* learning needs. To classify and prioritize the system's functional requirements, the MoSCoW Method approach was used, which divides requirements into four main categories. (I. Sommerville, 2015).

The first category is **"Must Have"** comprising the core features that must be available for the system to function and fulfill its main development objective. These crucial features included a login and registration system for user authentication, an Arabic word input field with validation to ensure only Arabic text is entered, and the integration of the Qutrub API for generating verb conjugation forms and the DeepSeek AI API for translations. Furthermore, the system was required to display structured conjugation results including past tense (*māḍī*), present tense (*muḍāriʿ*), and imperative (*amr*), show the *lughowi* conjugation form results for all pronouns (*dhomir*), and store the word search history based on the user's account.

Next, the **"Should Have"** category identified important features to support user comfort and experience, although not crucial to the main function. These features included a re-search

function from the search history, a responsive interface for mobile and desktop devices, notifications for when no results are found, a button to delete search history, and an admin dashboard to manage user activity.

The "**Could Have**" category contained additional features that could increase the system's educational value and flexibility, depending on time and resource availability. Examples included a dark mode for visual comfort, statistics on the most frequently searched words, a bookmark feature to save favorite words, auto-transliteration to help beginners read, and a comment or discussion feature for each word.

Finally, the "**Won't Have**" category identified features that were not planned for the initial version due to a lack of priority or time constraints. These excluded features were a PDF download function for search results, different user access levels for teachers and students, voice or audio pronunciation integration, and an open discussion forum between users. Using this method allowed for a focused and gradual system development process.

4.2 Design

In the design stage, interactions between the user and system are modelled using UML diagrams. One of the diagrams used is the Use Case Diagram to describe the system's functionality from the user's perspective.

The determination of the system's functional requirements depicted in this use case diagram was conducted using several methods, including the literature study method. The resulting model of user interactions with the *ArabicMorph* system are shown in **Figure 2**.

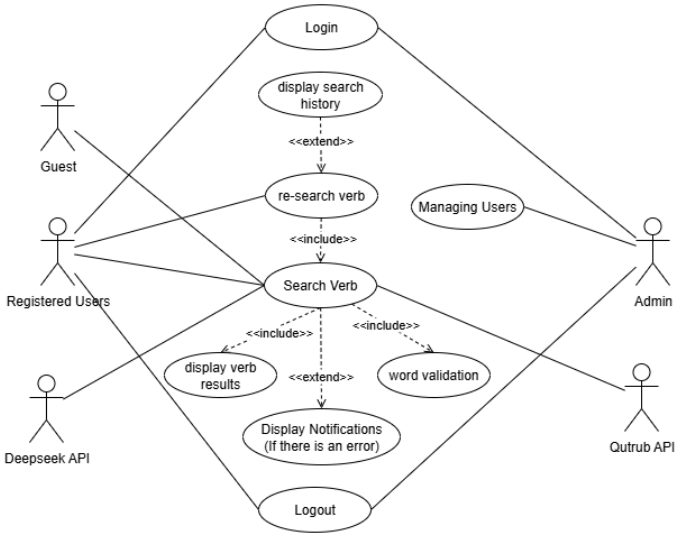

Figure 2. Use Case Diagram *ArabicMorph.*

The use case diagram in Figure 2 illustrates the interactions between five main actors—Guest, Registered User, Admin, DeepSeek API, and Qutrub API—in the *ArabicMorph* system. All actors, including guests, can search for Arabic verbs using the "Search Verb" feature, which is the core of the system. This feature includes input validation, calling external APIs (DeepSeek and Qutrub), and displaying results or notifications in case of errors. Registered users have additional features such as viewing and re-searching search history, which is an extension of the main search activity. Admins have special access to manage user data. All users can log in and log out as part of the system's access control. This use case emphasizes the integration of external services and access differentiation based on user roles.

An activity diagram is a way to model the events that occur within a use case. **Figure 3** is an illustration of the activity diagram related to the system process in operation.

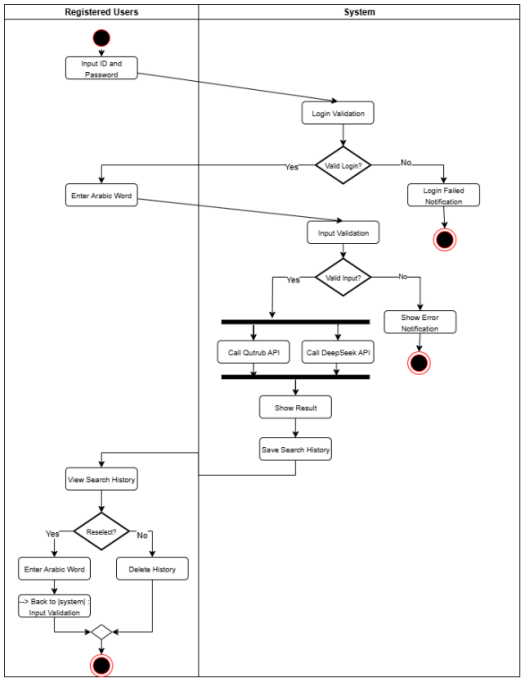

Figure 3. Activity Diagram Conjugation Registered Users.

Figure 3 shows the process of conjugating words and translating them with registered users as actors.

The main components of ERD are entities, attributes, and relationships. Entities are individuals that represent something real that can be distinguished from others. Relationships are connections between a number of entities that originate from different entities. The Entity Relationship Diagram in the design of the *ArabicMorph* website are shown in **Figure 4**.

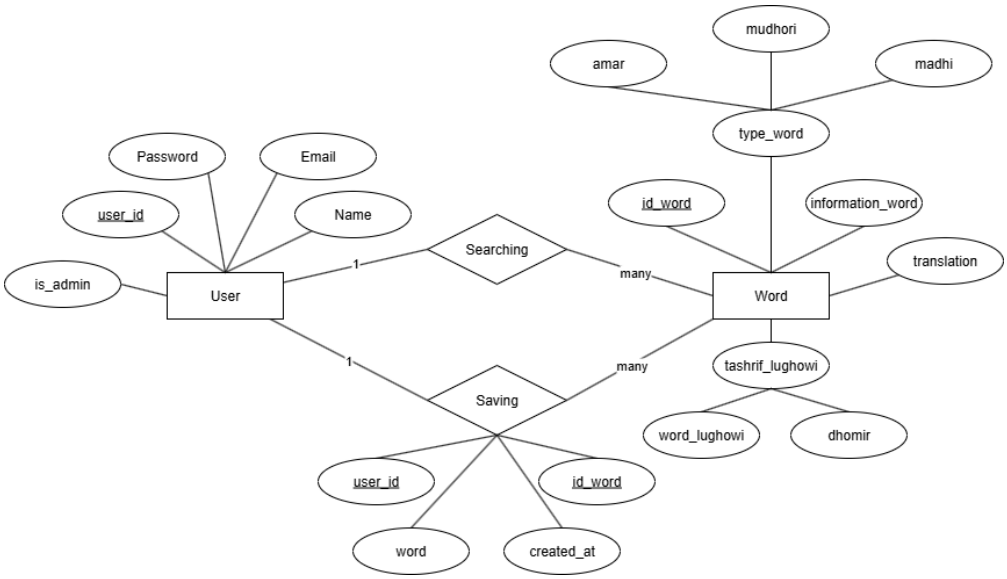

Figure 4. Entity Relationship Diagram of *ArabicMorph.*

This ERD (Entity Relationship Diagram) illustrates the database structure for the *sharf* system. There are three main entities: User, Word, and *tashrif_lughowi*. The User entity stores account information such as name, email, password, and the is_admin attribute to distinguish between administrators and regular users. Each user can search for multiple Words, which include verb information (past tense, present tense, imperative), word type, and translations. Additionally, users can save search results in the *tashrif_lughowi* entity, which contains word forms for various pronouns. The many-to-many relationship between users and words is modeled through search and save relationships, enabling the system to track user interactions with both processed and saved words.

A class diagram is a type of structural diagram in UML that clearly illustrates the structure and description of classes, attributes, methods, and relationships between objects. The class diagram for the *ArabicMorph* website design is shown in **Figure 5**.

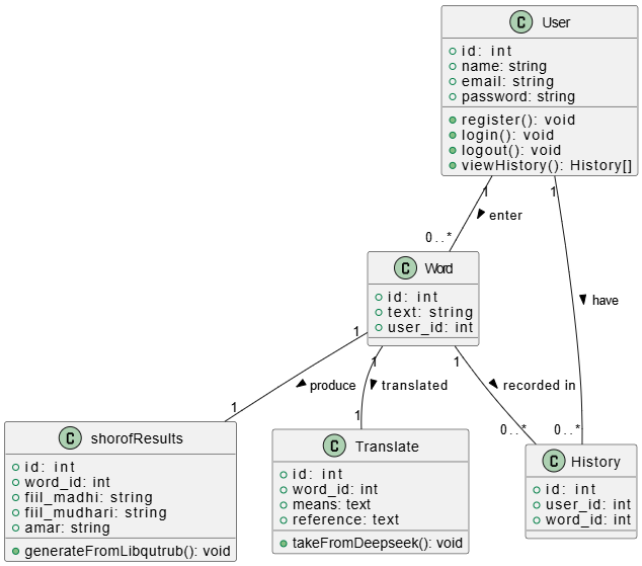

Figure 5. Class Diagram of *ArabicMorph.*

The class diagram of this *sharf* system illustrates the relationships between five main entities: **User**, **Word**, **Shorof Result**, **Translation**, and **History**. Users can register accounts, log in, and view search history. Each user can enter multiple words (Word), which are then processed to generate conjugation forms (Shorof Result) and translations (Translation) through integration with the Qutrub and DeepSeek APIs. Every search performed by a user is recorded in History, linking the user and the word. This diagram illustrates a structured data flow that supports personalized search functionality while remaining efficient for integrating *sharf* and automatic translation services.

4.3 Code (Implementation)

The user interface page contains files that can be accessed by administrators or users to access the admin dashboard, run the word search feature, and view the search history. The following is an explanation of the user interface that the author can provide

4.3.1 Main Page Display (Registered Users)

When a registered user successfully logs into the system, they are greeted with the main page, which serves as the primary dashboard for accessing the application's features. The interface for this main page is shown in **Figure 6**.

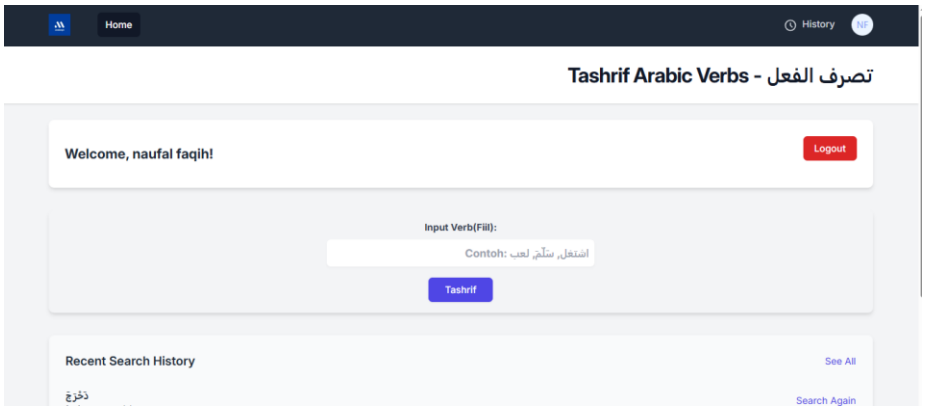

Figure 6. Main Page (Registered Users).

This page is the main menu display when a registered user successfully logs in.

4.3.2 Search Results Page Display

Upon entering a valid Arabic verb, the user is presented with the search result page. This interface, depicted in **Figure 7**, displays a summary of the word search, detailed information about the verb, and its translation.

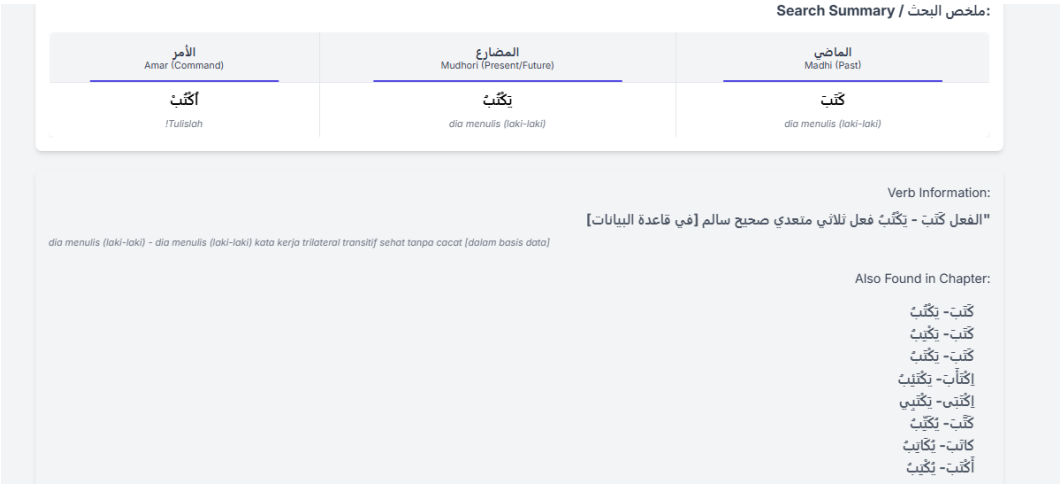

Figure 7. Search Results Page Display.

Figure 7 shows the page displayed as a result of a word entered by a user or guest, showing a summary of the word search, information about the word, and its translation.

4.3.3 Results Page Display (Lughowi)

For a more detailed morphological analysis, the system also displays the results of the word's application to the 14 types of pronouns (*Dhomir*). This comprehensive *lughowi* conjugation view is presented in **Figure 8**.

الضمائر	الماضي المعلوم	المضارع المعلوم	المضارع المجزوم	المض
أنا	كَتَبْتُ	أَكْتُبُ	أَكْتُبْ	أَكْتُبَ
نحن	كَتَبْنَا	نَكْتُبُ	نَكْتُبْ	نَكْتُبَ
أنت	كَتَبْتَ	تَكْتُبُ	تَكْتُبْ	تَكْتُبَ
أنتِ	كَتَبْتِ	تَكْتُبِينَ	تَكْتُبِي	تَكْتُبِي
أنتما	كَتَبْتُمَا	تَكْتُبَانِ	تَكْتُبَا	تَكْتُبَا
أنتما مؤ	كَتَبْتُمَا	تَكْتُبَانِ	تَكْتُبَا	تَكْتُبَا
أنتم	كَتَبْتُمْ	تَكْتُبُونَ	تَكْتُبُوا	تَكْتُبُوا
أنتن	كَتَبْتُنَّ	تَكْتُبْنَ	تَكْتُبْنَ	تَكْتُبْنَ
هو	كَتَبَ	يَكْتُبُ	يَكْتُبْ	يَكْتُبَ
هي	كَتَبَتْ	تَكْتُبُ	تَكْتُبْ	تَكْتُبَ
هما	كَتَبَا	يَكْتُبَانِ	يَكْتُبَا	يَكْتُبَا
هما مؤ	كَتَبَتَا	تَكْتُبَانِ	تَكْتُبَا	تَكْتُبَا
هم	كَتَبُوا	يَكْتُبُونَ	يَكْتُبُوا	يَكْتُبُوا
هن	كَتَبْنَ	يَكْتُبْنَ	يَكْتُبْنَ	يَكْتُبْنَ

Figure 8. Search Results Page Display (*Lughowi*).

Figure 8 shows the results of a search for words in their application to 14 types of pronouns (*Dhomir*).

4.4 Testing

The results of testing the functions in the system are shown in **Table 1** and **Table 2**. Meanwhile, **Table 2** displays the results of testing the functions in the registered user and Guest sections.

Table 1. Results of testing system functions in the admin section.

Module Test	Testing Procedures	Input	Expected Output	Results
Login admin	Enter email and password on the login page	Valid email and password	Admin is directed to the dashboard page	Valid
Manage Users	Access the “manage user” menu then click “make admin”	Add admin role	Account status changes and appears as admin	Valid

Table 2. Results of testing system functions in the user section.

Module Test	Testing Procedures	Input	Expected Output	Results
Word Search	Users enter Arabic verbs in the search field	Word: كَتَبَ	The system displays tashrif and translation results from the Qutrub and DeepSeek APIs.	Valid
Word Search (Invalid)	Guest entered non-Arabic characters	Input: "123abc"	The system displays a validation message: “Only Arabic text is allowed.”	valid
View History	Registered users log in and access the History menu.	Click the History menu	The system displays a list of previous searches.	Valid
Re-Search from History	The user clicks on an entry in the search history	Click on the entry "كَتَبَ"	The system displays the tashrif results and translation for the word.	Valid
Clear History	Registered user deletes one of the items in history	Click the delete button on the entry	The selected history is missing from the list.	Valid

In addition to alpha testing, beta testing was conducted by filling out questionnaires on the user side to evaluate the system that had been built. The results of this questionnaire are presented in Table 3. User responses consist of statements indicating dissatisfaction (TP), dissatisfaction (K), satisfaction (C), satisfaction (P), or high satisfaction (SP), with weight values ranging from 1 to 5 respectively. The number of respondents who provided evaluations was 17, with ages ranging from 15 to 22 years. The result of data questionnaire are shown in **Table 3**.

Table 3. Results of the questionnaire testing the system that has been developed.

Question	TP	K	C	P	SP	%
How satisfied are you with the word search feature?	0	0	2	5	10	89.4%
How satisfied are you with the results of the displayed shorof form?	0	0	2	6	9	88.2%
How satisfied are you with the translations provided from DeepSeek?	0	0	1	4	12	93%
How satisfied are you with the overall appearance of this application?	0	0	3	3	11	89.4%
The Arabic word search feature is very helpful for users.	0	0	0	6	11	93%
The system displays accurate shorof form information.	0	0	0	7	10	91.8%

I found the references from DeepSeek helpful in understanding the meaning of the word.	0	0	0	7	10	91.8%
This application is suitable for learning the science of shorof independently.	0	0	0	5	12	94.2%
How satisfied are you with the overall ease of use of this application?	0	0	1	3	13	94.2%
Overall Average						91.8%

Based on the results of beta testing through questionnaires, the average user satisfaction and approval rating was 91.8%, indicating that the system received a very positive response. All aspects tested—from search features, shorof results, DeepSeek translations, to ease of use—received a majority of “Satisfied” and “Very Satisfied” responses, with no negative feedback (TP or K). This indicates that the system successfully meets user expectations, both in terms of functionality and usability. Therefore, the system can be considered suitable and ready for broader use, with minor improvements based on user feedback.

5. Conclusion

The *ArabicMorph* website was successfully designed and implemented as a web-based digital solution for learning Arabic grammar, combining the Qutrub API for Arabic verb conjugation and DeepSeek AI for automatic translation. All the Must-Haves defined through MoSCoW needs analysis—from user authentication, validated Arabic word input, API calls, to history storage—have been successfully implemented. An evaluation via a questionnaire among 17 respondents revealed an average satisfaction rate of 91.8%, indicating the application is highly regarded as helpful, accurate, and user-friendly. These results demonstrate that the integration of rule-based linguistic technology and AI within the Laravel interface can address the challenges of learning sharf, particularly for students who struggle with memorizing word change patterns.

References

Anwar, M. (2019). *Ilmu Sharf : Terjemaahan Matan Kailani dan Nazham Almaqsud* (pp. 58–62). Sinar Baru Algensindo.

Booch, G., Rumbaugh, J., & Jacobson, I. (2017). Unified Modeling Language User Guide , The (2nd Edition) (Addison-Wesley Object Technology Series) Unified Modeling Language User Guide , The Unified Modeling Language User Guide , The Many of the designations used by manufacturers and sellers to dist. In *ResearchGate* (Vol. 2nd, Issue August).

Dubois, P. (2013). *MySQL Developer’s Library* (fifth edit). Pearson Education.

I. Sommerville. (2015). Software Engineering. In *Software Engineering: Barry W. Boehm’S Lifetime Contributions to Software Development, Management, and Research* (10th ed.). Pearson Education. https://doi.org/10.1109/9780470187562.ch8

M., L. R. and M. A. (2013). *RESTful Web APIs*. O’Reilly Media.

Pressman, R. S., & Maxim, B. R. (2015). Software Engineering A Practitioner’s Approach. In *Proceedings of the National Academy of Sciences* (Vol. 3, Issue 1). http://dx.doi.org/10.1016/j.bpj.2015.06.056%0Ahttps://academic.oup.com/bioinformatics/article-abstract/34/13/2201/4852827%0Ainternal-pdf://semisupervised-3254828305/semisupervised.ppt%0Ahttp://dx.doi.org/10.1016/j.str.2013.02.005%0Ahttp://dx.do

i.org/10.10
Stauffer, M. (2023). *Laravel: Up & Running THIRD EDITION A Framework for Building Modern PHP Apps*. http://oreilly.com
Sukamto, Rosa Ariani & Salahuddin, M. (2016). *Rekayasa perangkat lunak terstruktur dan berorientasi objek*.
Yunisa, M. (2022). Problematika Pembelajaran Bahasa Arab dalam Aspek Ilmu Nahwu dan *Sharf* pada Siswa Kelas X Madrasah Aliyah Laboratorium Jambi. *Jurnal Pendidikan Bahasa Arab Dan Budaya Islam*, *03*(2), 6.
Zerrouki, T. (2020). *Towards An Open Platform For Arabic Language Processing*. *July*. https://doi.org/10.13140/RG.2.2.29882.82881