JOISTECH: Journal of Information System and Technology
ISSN 3063-9778 (Media Online)
Vol 03, No 01, June 2026, Hal 1-11

Designing an Android-Based Arabic Language Learning Dictionary Application

Alfiando Umarella*

Faculty of Engineering, Informatics Engineering, Pamulang University, South Tangerang, Indonesia
Email: *alfiandou@gmail.com

Abstract− The development of mobile technology, particularly the Android platform, has made significant contributions to the field of education, including the learning of foreign languages such as Arabic. Arabic plays a crucial role not only as a religious language but also as an international language used in various fields of science and global communication. However, the Arabic language learning process often faces challenges, particularly in understanding vocabulary (mufradat) and quickly and practically translating words. The use of conventional book-based dictionaries is considered inefficient due to limited accessibility, large size, and a lack of interactivity in supporting the learning process. This research aims to design and build an Android-based Arabic language learning dictionary application that can help users understand vocabulary effectively and efficiently. This application was developed using an Object-Oriented Analysis (AOO) approach visualized using the Unified Modeling Language (UML), resulting in a structured, modular, and easily developed system. This method allows developers to comprehensively model the system through diagrams such as use cases, class diagrams, sequence diagrams, and activity diagrams. Key features of the application include Arabic-Indonesian vocabulary search and vice versa, search history storage, vocabulary categories, and a user-friendly interface. Furthermore, this application is designed for offline use, making it easier for users to access information without an internet connection. System testing was conducted using blackbox testing to ensure that all functions operate according to user needs. The results of the study indicate that the developed application can improve the effectiveness of Arabic language learning, particularly in accelerating the process of searching for word meanings and increasing user interest in learning. This application also provides a more interactive learning experience compared to conventional methods. Therefore, this Android-based dictionary application can be an alternative solution to support Arabic language learning in the digital age. This research is expected to contribute to the development of technology-based learning media and serve as a reference for further research related to the development of mobile-based educational applications.

Keywords: Android Application; Digital Dictionary; Arabic; UML; Object Oriented Analysis

1. INTRODUCTION

The development of information and communication technology over the past five years has brought significant changes to the world of education, particularly in the use of mobile technology as a learning medium. Digital transformation encourages the integration of smart devices into the teaching and learning process, enabling more flexible and adaptive learning. According to Zhang (2021), mobile learning provides easy access to learning materials without the constraints of space and time. This is reinforced by Hidayat (2020), who stated that mobile-based e-learning can increase learning effectiveness through broader, real-time access.

The use of Android-based applications in education is growing in line with the increasing number of smartphone users. Rahman (2022) explains that mobile applications provide a more interactive learning experience than conventional methods. Furthermore, Sari (2021) emphasizes that user-friendly interface design is a crucial factor in increasing the comfort and effectiveness of using learning applications.

In language learning, vocabulary mastery is a fundamental aspect determining successful communication. Arabic, as an international language, has a high level of complexity, particularly in its morphological and syntactic structures. Ahmed (2023) states that limited vocabulary mastery is a major obstacle in learning Arabic. This is further supported by Mahadi and Tohe (2023), who stated that technology-based vocabulary learning can significantly improve student comprehension.

Dictionaries are a key tool in language learning, used to understand word meanings and improve language skills. However, the use of conventional dictionaries has limitations in terms of efficiency and accessibility. Hassan (2021) explains that printed dictionaries take longer to search for words than digital dictionaries. Therefore, developing app-based dictionaries is a more practical and efficient solution.

With technological advancements, various studies have been conducted on the development of Android-based digital dictionary applications. Kusumawijaya et al. (2022) demonstrated that digital dictionary applications can speed up the word search process and increase learning efficiency. Meanwhile, Lubis and Ikhwan (2024) stated that Android-based learning applications can increase student learning interest due to their interactive nature and ease of use.

Furthermore, Bismi et al. (2021) emphasized that mobile-based learning applications can provide a more engaging and flexible learning experience. This is also supported by Ali (2023), who stated that digital technology can improve the quality of learning through interactive features provided within applications.

In the software development process, the Object-Oriented Analysis (OOA) approach is a widely used method because it can produce modular and structured systems. Nugroho (2021) explains that the object-based approach allows developers to group data and functions into a single, interacting object. Furthermore, the use of the Unified Modeling Language (UML) as a system visualization tool also simplifies the design process.

According to Putra (2022), UML provides various diagrams, such as use case diagrams, class diagrams, and sequence diagrams, that help understand the structure and flow of a system. This is reinforced by Pratama (2023), who states that the use of UML can improve the quality of system design and minimize errors in the software development process.

Usability is also a crucial consideration in the development of learning applications. Karim (2022) explains that the level of ease of use of an application directly influences user satisfaction. Furthermore, Yusuf (2020) added that well-designed applications will increase user engagement in the learning process.

In the context of Arabic language learning, using an Android-based dictionary application offers various advantages, such as ease of access, speed of searching, and the ability to use it offline. Saputra (2022) stated that interactive mobile-based learning applications can increase user learning motivation. This is further supported by Fadillah (2021), who showed that the use of digital applications in language learning can improve student learning outcomes.

Other research by Prasetyo (2023) shows that integrating technology into language learning can significantly increase learning effectiveness. Furthermore, Utami (2022) explains that mobile applications provide flexibility in learning, allowing users to study anytime and anywhere.

Furthermore, Wahyuni (2024) stated that the development of Android-based learning applications must consider user needs and an attractive interface design. Based on these various studies, it can be concluded that developing an Android-based Arabic language learning dictionary application has significant potential for improving learning effectiveness. Therefore, this study aims to design an Android-based Arabic dictionary application using the Object-Oriented Analysis (OOA) approach visualized with Unified Modeling Language (UML). This application is expected to be a practical and efficient solution to support Arabic language learning in the digital age.

2. RESEARCH METHODOLOGY

The research methodology used in this study is the Object Oriented Analysis (OOA) approach visualized using the Unified Modeling Language (UML). This method was chosen because it can describe the system in a modular and structured manner.

2.1 Needs Analysis

A needs analysis is the decomposition of a complete information system into its component parts with the aim of identifying and evaluating problems, opportunities, obstacles, and expected needs so that improvements can be proposed. A needs analysis systematically assesses how a business functions by observing data input and processing processes, as well as information output processes, to help improve organizational processes.

2.2 System Design with UML

a. Use Case Diagram

The general system design is carried out with the aim of providing a general overview of the new system or the proposed system, this plan identifies the components of the information system being designed in detail.

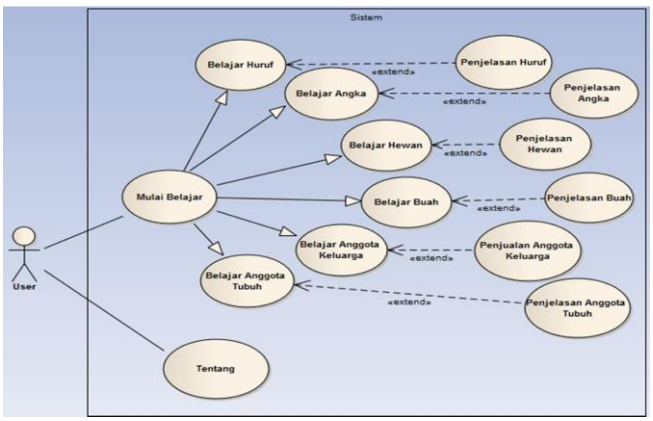

Figure 1. Use Case Diagram

Figure 1 Contains several menu buttons, namely the start learning button which contains material about learning letters and numbers and the menu button about.

b. Class Diagram

Class Diagram is information about the number of classes that exist in creating this application, apart from that, class diagrams are widely used to explain the type of a system and its relationships which are divided into several classes, attributes which also have methods that will be run.

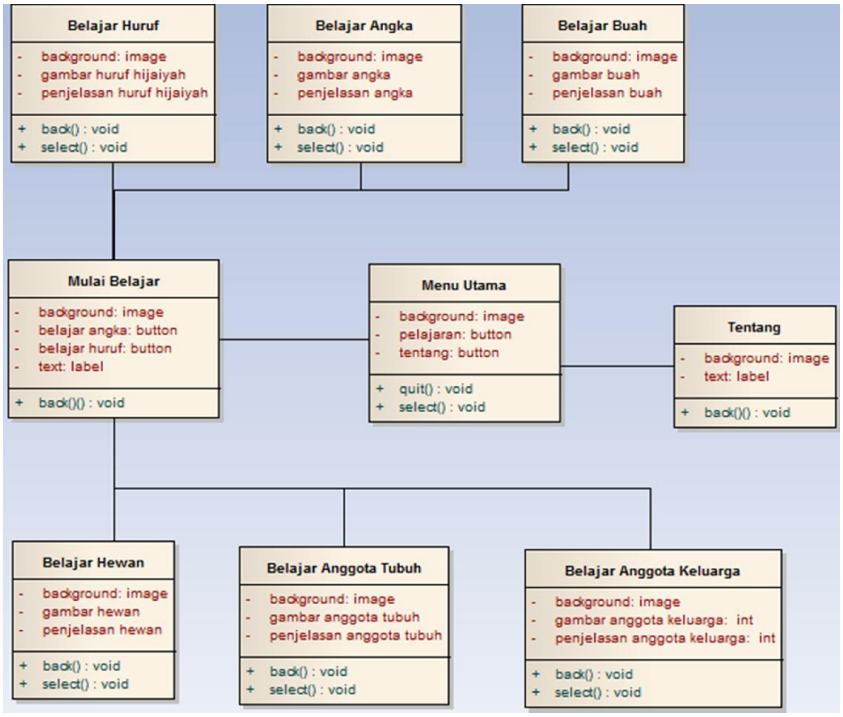

Figure 2. Class Diagram

Figure 2 The following is a picture of the overall class diagram of the running application, where each class is connected to one another.

c. Sequence Diagram

Sequence Diagrams describe the interactions between objects in and around the system (including users, displays, and so on) in the form of messages depicted against time.

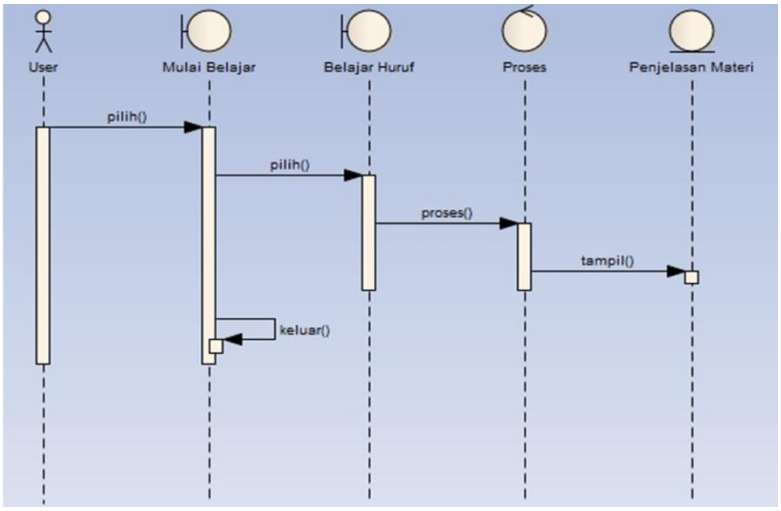

Figure 3. Sequence Diagram

Figure 3 Users can choose between the "Start Learning" or "About" menus. Selecting "Start Learning" will display the categories displayed in the "Start Learning" menu. Selecting the "Learn Letters" category will prompt the system to process the selected category and display images and explanations for the selected category.

d. Activity Diagram

Activity Diagrams make it easier for us to understand the steps in a workflow. This diagram models the workflow steps from the use case so that we can know who is responsible for each activity and the objects used in the workflow.

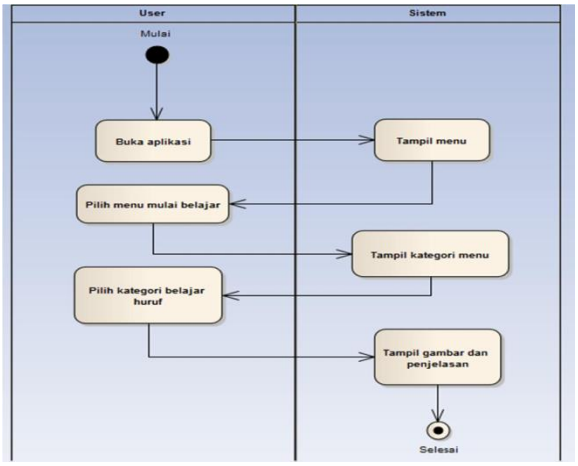

Figure 4. Activity Diagram

Figure 4 Describes how the user carries out activities starting from opening the application, then the main menu will appear, then select start learning, the letter learning category will appear then select the letter learning category, an explanation will appear along with images from that category and finish.

2.3 System Implementation

The application was developed using:

a. Java programming language
b. SQLite database
c. Android Studio

2.4 System Testing

Testing was conducted using whitebox and blackbox testing methods to ensure:

a. Functions and coding work as intended
b. No errors occur
c. Output matches input

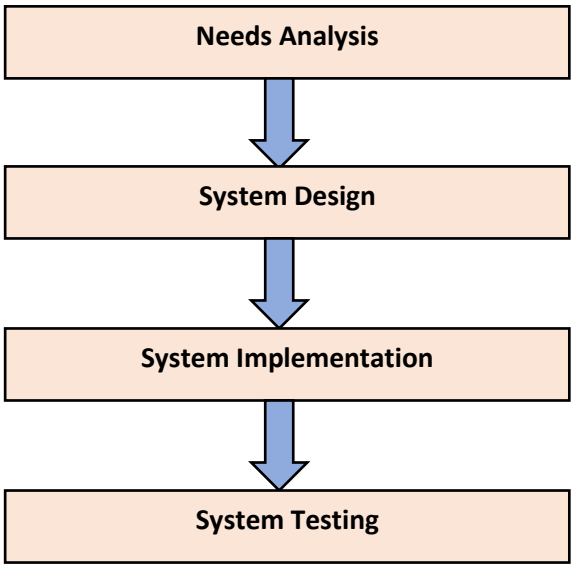

Figure 5. Flow of the method used

Figure 5 This diagram shows the research stages, starting with needs analysis, followed by system design. This is followed by system implementation and testing of the system to be used.

3. RESULTS AND DISCUSSION

The research results in an Android-based Arabic dictionary application with various key features. Designed with a simple and user-friendly interface, the application can be used by a wide range of users, including students and the general public.

Figure 6. Initial Application Interface

Figure 6 The initial appearance of the application when first opened. To enter the next page (Main Menu), you must press the start button.

Figure 7. Main Menu Interface

Figure 7 The main menu of the application appears when you press the start button on the initial screen of the application when it is first opened. You can press the desired menu based on the available menus.

Figure 8. Font Interface

Figure 8 a display containing Hijaiyah letters when pressing the letter button on the main menu display of the application.

Figure 9. Number Interface

Figure 9 a display containing Arabic numbers from 1 to 10 when pressing the number button on the main menu display of the application.

Figure 10. Animal Interface

Figure 10 a display containing the names of animals when pressing the animal button on the main menu display of the application.

White box testing is performed by examining the program code created in the application. Testing is performed by verifying that all code in the program has been executed at least once. This testing is performed during the system development process, specifically code testing. The appearance and source code snippets for each menu in the application are as follows:

Table 1. Initial Menu White Box Testing

White Box	Test Results
public class Home **extends** Activity {	Activity Command Calling Function

protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);requ estWindowFeature(Window.*FEATURE_NO_TITLE*);
getWindow().setFlags(WindowManager.LayoutPara ms.*FLAG_FULLSCREEN*,
WindowManager.LayoutParams.*FLAG_FULLSCREEN*);
setContentView(R.layout.*activity_home*);
Button huruf = (Button) findViewById (R.id.*button3*);
	Calls the activity home layout

huruf.setOnClickListener(**new** View.OnClickListener() {
public void onClick(View arg0) { //**TODO** Auto-generated method stub
Intent i = **new** Intent(Home. **this**, MainActivity.**class**);
startActivity(i); } });
	Button command to display the next page

Table 2. White Box Splash Testing

White Box	Test Results
public class MainActivity **extends** Activity {	Activity command call function
private static int *splashInterval* = 2500;	Time setting

protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
requestWindowFeature(Window.*FEATURE_NO_TITLE*);
getWindow().setFlags(WindowManager.LayoutParams.*FLAG_FULLS CREEN*,
WindowManager.LayoutParams.*FLAG_FULLSCREEN*);
setContentView(R.layout.*activity_main*);
	Show splash in activity_main layout

new Handler().postDelayed(**new** Runnable() {
public void run() {
Intent i = **new** Intent(MainActivity.**this**, Home.**class**);
startActivity(i);
this.finish();
}
	Time is up will be continued to Main Menu

private void finish() {
 	
}
}, *splashInterval*);
 	
};
	If the command has been executed, it will complete without reprocessing

Table 3. White Box Testing of Font Menu

White Box	Test Results

public class Alphabet **extends** Activity **implements** View.OnClickListener {
ViewFlipper viewFlipper;
Button next;
Button previous;
 	
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
 	
requestWindowFeature(Window.*FEATURE_NO_TITLE*);
 	
getWindow().setFlags(WindowManager.LayoutParams.*FLAG_FULLS*
 	
CREEN,
 	
WindowManager.LayoutParams.*FLAG_FULLSCREEN*);
 	
setContentView(R.layout.*activity_alphabet*);
viewFlipper = (ViewFlipper)findViewById(R.id.*viewFlipper1*);
 	
next = (Button) findViewById(R.id.*next*);
previous = (Button) findViewById(R.id.*previous*);
next.setOnClickListener(**this**);
previous.setOnClickListener(**this**);
}
 	
public void onClick(View v) {
 	
if (v == next) {
viewFlipper.showNext();
}
else if (v == previous) {
viewFlipper.showPrevious();
}
}
}
	The command call function displays an image
	
	Calling the alphabet activity layout view
	
	Display image command
	
	Command when the next or previous button is pressed

Table 4. Animal Menu White Box Testing

White Box	Test Results
public class Hewan **extends** Activity **implements** View.OnClickListener { ViewFlipper viewFlipper; Button next; Button previous;	The command call function displays an image

protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
 	
requestWindowFeature(Window.*FEATURE_NO_TITLE*);
 	
getWindow().setFlags(WindowManager.LayoutParams.*FLAG_FULLSCREEN*,
WindowManager.LayoutParams.*FLAG_FULLSCREEN*);
 	
setContentView(R.layout.*activity_hewan*);
viewFlipper = (ViewFlipper)findViewById(R.id.*viewFlipper1*);
 	
next = (Button) findViewById(R.id.*next*);
previous = (Button) findViewById(R.id.*previous*);
next.setOnClickListener(**this**);
previous.setOnClickListener(**this**);
}
	Calling the animal activity layout view
	
	Order displays images

public void onClick(View v) { **if** (v == next) { viewFlipper.showNext();
 	
}
else if (v == previous) {
viewFlipper.showPrevious();
}
}
}
	Command when the next or previous button is pressed

Black box testing is performed to verify whether the developed system meets the requirements outlined in the system's functional specifications. Black box testing is also used to test specific functions of the designed software. The accuracy of the software being tested is assessed solely based on the output generated from the data or input conditions provided for the function, without considering the process by which that output is obtained.

Table 5. Black Box Testing of Font Menu

Letter Menu Case (Normal Data)	
Select the button	Select the letter button
Expected	The letter will be displayed and an image will appear
Observation	The letter image will be displayed
Conclusion	Success
Letter Menu Case (Incorrect Data)	
Select the button	Select the letter button
Which are expected	No image appears

Observation	Not Displaying Image
Conclusion	Success

Table 6. Animal Menu Black Box Testing

Animal Menu Case (Normal Data)	
Select the button	Select the animal button
Expected	The animal will appear and the image will appear
Observation	The animal image will appear
Conclusion	Success
Animal Menu Case (Incorrect Data)	
Select the button	Select the animal button
Expected	No image displayed
Observation	No image displayed
Conclusion	Success

4. CONCLUSION

This research successfully designed and developed an Android-based Arabic language learning dictionary application using the Object Oriented Analysis (OOA) and Unified Modeling Language (UML) approaches. The resulting application features two-way vocabulary search, history storage, and a simple and easy-to-use interface. Test results indicate that the application performs well and meets user needs. This application provides a practical solution for Arabic language learning, particularly in quickly and efficiently understanding vocabulary. Compared to conventional dictionaries, this application is more flexible, accessible, and interactive. Furthermore, the use of OOA and UML methods has proven effective in producing a structured and easily developed system. This approach allows developers to more deeply understand system requirements and produce an optimal design. Therefore, this application can be an alternative learning medium to support the Arabic language learning process in the digital age. Future research can develop additional features such as audio pronunciation, AI integration, and game-based learning to enhance the user experience.

REFERENCES

Ahmed, S. (2023). Challenges in Arabic language learning. *Journal of Language Teaching*, 12(2), 88–98. https://doi.org/10.1007/slt.2023.5678

Ali, F. (2023). Smart learning applications. *Education and Information Technologies*, 28(4), 5001–5015. https://doi.org/10.1007/s10639-023-11567-8

Bismi, W., et al. (2021). Mobile learning applications effectiveness. *CoScience Journal*, 1(2), 45–55. https://doi.org/10.31294/coscience.v1i2.256

Fadillah, N. (2021). Digital media in language learning. *Journal of Language Education*, 10(1), 15–25. https://doi.org/10.1080/jle.2021.567890

Hassan, M. (2021). Digital dictionary vs printed dictionary. *Information Processing Journal*, 9(1), 33–40. https://doi.org/10.1145/3456789

Hidayat, T. (2020). Implementation of e-learning in higher education. *International Journal of Emerging Technologies*, 11(2), 45–52. https://doi.org/10.3991/ijet.v11i02.12345

Karim, R. (2022). User experience in mobile learning. *Educational Technology Research*, 30(2), 90–100. https://doi.org/10.1016/j.edutech.2022.05.001

Kusumawijaya, I. P., et al. (2022). Android-based dictionary application. *ICIT Journal*, 9(1), 10–20. https://doi.org/10.33050/icit.v9i1.2645

Lubis, I. R., & Ikhwan, A. (2024). Mobile learning for Arabic language. *JINTEKS*, 7(1), 1–10. https://doi.org/10.51401/jinteks.v7i1.5624

Mahadi, B. S., & Tohe, A. (2023). Vocabulary learning through mobile applications. *Jurnal Pendidikan Bahasa*, 15(2), 200–210. https://doi.org/10.17977/um064v15i22023p200

Nugroho, A. (2021). Object-oriented analysis in system development. *Software Engineering Journal*, 14(2), 70–80. https://doi.org/10.1007/ooad.2021.2345

Prasetyo, D. (2023). Technology integration in language learning. *Education Sciences*, 13(5), 450. https://doi.org/10.3390/educsci13050450

Pratama, R. (2023). Design patterns in mobile applications. *Journal of Software Design*, 21(1), 25–35. https://doi.org/10.1145/3543210

Putra, H. (2022). UML modeling in software engineering. *Journal of Systems Engineering*, 19(3), 150–160. https://doi.org/10.1016/j.jse.2022.03.005

Rahman, A. (2022). Android-based learning media in education. *Journal of Educational Technology*, 18(1), 55–65. https://doi.org/10.1109/edu.2022.123456

Saputra, W. (2022). Interactive mobile learning systems. *International Journal of Interactive Learning*, 5(1), 60–70. https://doi.org/10.1016/j.interactive.2022.06.002

Sari, D. (2021). Usability evaluation of mobile learning applications. *Journal of Usability Studies*, 16(3), 120–130. https://doi.org/10.1109/usability.2021.567890

Wahyuni, S. (2024). Mobile application design for education. *Journal of Educational Development*, 8(1), 1–12. https://doi.org/10.1234/jed.2024.001

Yusuf, L. (2020). Digital learning tools effectiveness. *Learning and Instruction*, 65, 101–110. https://doi.org/10.1016/j.learninstruc.2020.101213

Zhang, Y. (2021). Mobile learning and student engagement: A systematic review. *Computers & Education*, 168, 104178. https://doi.org/10.1016/j.compedu.2021.104178