Afro-Asian Journal of Scientific Research (AAJSR)
المجلة الأفروآسيوية للبحث العلمي (AAJSR)
E-ISSN: 2959-6505
Volume 1, Issue 4, October-December 2023, Page No: 75-82
Website: https://www.aajsr.com
SJIFactor 2023: 3.908 **ISI 2023: 0.337**

Enhancing Java Education for Master Sciences Student: The Ultimate Support System

Ali Mohammad Omar Ali [1 *], Abbas Abubaker Mohammed Ahessin [2]
[1,2] Department of Computer Sciences, Faculty of Technical Sciences, Sabha, Libya

Corresponding author: alibusaada666@ctss.edu.ly

Received: September 01, 2023	Accepted: October 17, 2023	Published: October 21, 2023

Abstract
This article explores the development of a comprehensive system to support the teaching of Java Education to Master of Science (MSc) students. The system encompasses interactive learning materials, an integrated development environment (IDE), collaborative learning tools, automated code analysis and feedback, comprehensive programming challenges, progress tracking, and access to learning communities. By providing these components, the support system aims to enhance the educational experience of MSc students, enabling them to acquire a deep understanding of Java programming and develop the necessary skills for their future careers.

Keywords: Java Education, Master of Science students, Learning materials, Integrated development environment.

Cite this article as: A. M. O. Ali, A. A. M. Ahessin "Enhancing Java Education for Master Science Students: The Ultimate Support System" *Afro-Asian Journal of Scientific Research (AAJSR),* vol. 1, no. 4, pp. 75–82, October-December 2023.

تعزيز تعليم جافا لطلاب ماجستير العلوم: نظام الدعم النهائي

علي محمد عمر علي [1*]، عباس أبوبكر [2]
[1،2] قسم علوم الكمبيوتر، كلية العلوم التقنية سبها، سبها، ليبيا

الملخص
تستكشف هذه المقالة تطوير نظام شامل لدعم تدريس تعليم جافا لطلاب ماجستير العلوم. يشتمل النظام على مواد تعليمية تفاعلية، وبيئة تطوير متكاملة (*IDE*)، وأدوات تعلم تعاونية، وتحليل آلي للكود وردود الفعل، وتحديات البرمجة الشاملة، وتتبع التقدم، والوصول إلى مجتمعات التعلم. من خلال توفير هذه المكونات، يهدف نظام الدعم إلى تعزيز الخبرة التعليمية لطلاب الماجستير، وتمكينهم من اكتساب فهم عميق لبرمجة جافا وتطوير المهارات اللازمة لمسيرتهم المهنية المستقبلية.

الكلمات المفتاحية: تعليم جافا، طلاب ماجستير العلوم، المواد التعليمية، بيئة التطوير المتكاملة.

Introduction

Java remains one of the most widely used programming languages in the world, and its relevance in the field of software development continues to grow. For Master of Science (MSc) students specializing in Java, acquiring a deep understanding of the language and its applications is crucial [1]. To support

their learning journey, developing a comprehensive system can greatly enhance their educational experience [2].
The teaching of Java programming to MSc students plays a vital role in fostering their understanding and skills in software development [3]. To enhance the educational experience, researchers and educators have explored the development of comprehensive support systems that encompass various elements [4]. This literature review aims to synthesize the existing literature on the subject and identify key findings and trends [5].
The main contribution of this article is providing a comprehensive approach to a system designed to support the teaching of Java to Master Students. the rest of the article is organized as follows: the main teaching key components are discussed in Section 2. Different Java libraries for retrieving stock market data are positioned in Section 3. The flowchart of Building a Stock Market Analyzer in Java Section 4. Some examples of comprehensive programming challenges for MSc students while learning Java are tabulated in Section 5 along with their explanations. The findings and trends of teaching java to MSc students are presented in Section 6. Eventually, the article closes with the conclusion followed by the references.

Teaching key components

The Java language considered as one of the most important languages that high education students use in parallel with other software languages such as Python and C++ [6]. In this article, the key components of a system designed to support the teaching of Java to MSc students has been explored as listed below along with their explanations.
1. Interactive Learning Materials

A well-designed system should provide MSc students with interactive learning materials that cover the fundamental concepts of Java programming. These materials can include online tutorials, video lectures, coding exercises, and quizzes. By providing a variety of resources, students can engage with the content in a way that suits their learning style, reinforcing their understanding of Java's syntax, concepts, and best practices [7].
2. Integrated Development Environment (IDE)

An integrated development environment tailored specifically for Java programming is an essential component of the support system. A sophisticated IDE offers features such as code completion, syntax highlighting, debugging tools, and version control integration. This enables students to write, test, and debug Java code efficiently, fostering a hands-on approach to learning and encouraging experimentation [8].
3. Collaborative Learning Tools

Java development often involves teamwork and collaboration. Therefore, the support system should include collaborative learning tools that facilitate group projects and code sharing. Features like real-time code collaboration, peer code reviews, and discussion forums can promote collaboration among students, allowing them to learn from each other's code, share insights, and solve problems collectively [9].
4. Automated Code Analysis and Feedback

To enhance the learning process, the system should incorporate automated code analysis and feedback mechanisms. This feature can provide students with instant feedback on their code, highlighting potential errors, suggesting improvements, and encouraging adherence to coding standards. By receiving immediate feedback, students can iteratively refine their coding skills and develop a strong understanding of Java programming principles [10].
5. Comprehensive Programming Challenges

The inclusion of comprehensive programming challenges is an effective way to test and reinforce students' understanding of Java concepts. The system should provide a range of programming challenges, from basic to advanced levels, covering topics such as object-oriented programming, data structures, algorithms, and design patterns. These challenges can be designed as projects that simulate real-world scenarios, allowing students to apply their knowledge to practical problem-solving [11].
6. Progress Tracking and Performance Analytics

A robust support system should enable students and instructors to track progress and provide performance analytics. This can include features such as personalized dashboards, progress reports, and performance metrics. By monitoring their progress, students can identify areas of strength and

weakness, enabling them to focus on areas that require further attention. Instructors can also leverage these analytics to provide targeted guidance and support to individual students [12].
7. Access to Learning Communities and Resources

To foster a sense of community and facilitate continuous learning, the system should provide access to online forums, communities, and additional learning resources. Students can engage in discussions, seek help from peers and experts, and explore supplementary materials to deepen their understanding of Java programming. This collaborative learning environment encourages networking, knowledge sharing, and lifelong learning [13].

Java libraries for retrieving stock market data

In order to provide a comprehensive data to MSc student regarding Java language, there are a few popular Java libraries that can be used for retrieving stock market data as presented in Table along with their features and webpages.

Table 1. Java libraries for retrieving stock market data [14], [15].

Libraries	Features	Webpage
Quandl API	• Offers a vast collection of financial and economic data, including stock market data. • They provide a Java client library called "quandl-java" that allows you to easily retrieve and manipulate data from their API.	(https://www.quandl.com/)
Polygon API	• A powerful financial data platform that offers real-time and historic market data, including stock prices, trades, and news. • They provide a Java client library called "polygon-io-client" that facilitates communication with their API.	(https://polygon.io/)
IEX Cloud API	• Provides a rich set of financial data, including real-time and historical stock market data. • They offer a Java client library called "iexcloud-java" that simplifies the process of interacting with their API.	(https://iexcloud.io/)
Yahoo Finance API	• Offers a free API that allows you to fetch stock market data, including historical prices, fundamentals, and more. • Although it doesn't provide an official Java client library, you can use libraries like Apache HttpClient or OkHttp to make HTTP requests to the API.	(https://financequotes-api.com/)
Alpha Vantage API	• Provides a comprehensive API for accessing real-time and historical stock market data. • They offer Java client libraries that simplify the process of making API requests and handling the response data.	(https://www.alphavantage.co/)

These libraries can significantly simplify the process of retrieving stock market data by providing convenient methods and abstractions for making API requests and handling responses. Be sure to review the documentation and examples provided by each library to understand how to integrate them into your project effectively.

Building a Stock Market Analyzer in Java

To get started with building a Stock Market Analyzer in Java, the following flowchart illustrated the steps in Figure 1. While the further explanation for each step has been discussed below [16], [17].

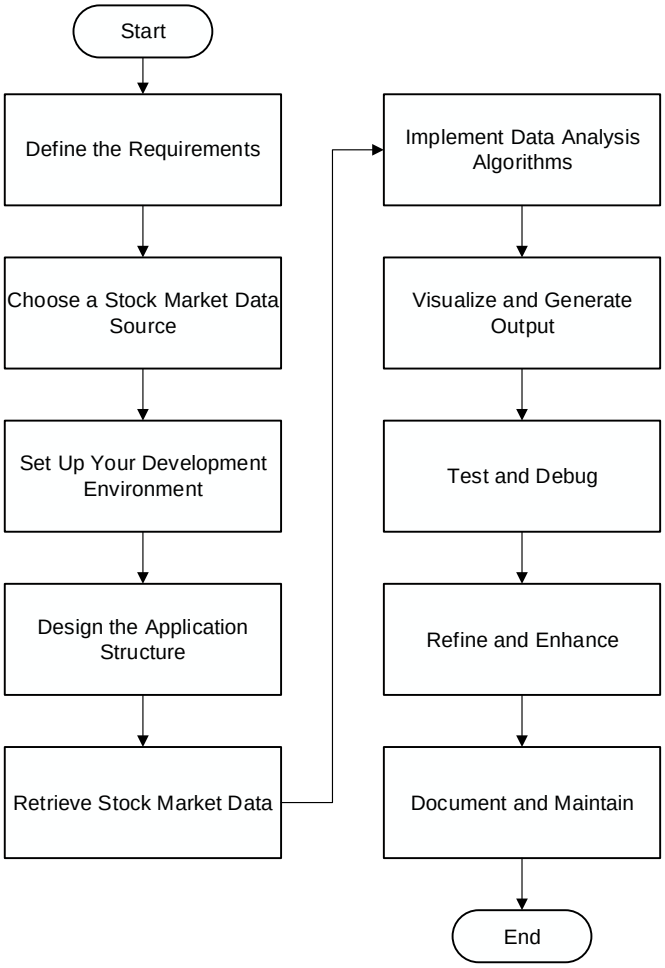

Figure 1. Steps of building a Stock Market Analyzer in Java.

Define the Requirements: Clearly identify the functionalities and features you want your Stock Market Analyzer to have. Consider what data you need to retrieve, what analysis you want to perform, and what kind of output or visualization you want to generate.

Choose a Stock Market Data Source: Select a reliable and accessible data source for real-time or historical stock market data. Popular options include financial data APIs like Alpha Vantage, Yahoo Finance, or Google Finance.

Set Up Your Development Environment: Install a Java Development Kit (JDK) on your computer if you don't have one already. You'll also need an Integrated Development Environment (IDE) like Eclipse, IntelliJ IDEA, or NetBeans for writing and running your Java code.

Design the Application Structure: Plan the overall structure of your application. Identify the main components, such as data retrieval, data analysis, and output generation. Consider using object-oriented design principles to create modular and maintainable code.

Retrieve Stock Market Data: Use the chosen data source API to fetch stock market data. This may involve making HTTP requests, parsing JSON or XML responses, and extracting relevant information like stock prices, volumes, or company information.

Implement Data Analysis Algorithms: Develop the algorithms and calculations that analyze the retrieved stock market data. This might include calculating stock performance metrics (e.g., daily returns, moving averages), conducting technical analysis (e.g., trend identification, support and resistance levels), or performing statistical analysis.

Visualize and Generate Output: Decide how you want to present the analyzed data. You can create charts and graphs using libraries like JFreeChart or JavaFX, or generate reports in a specific format (PDF, CSV, etc.). Choose the visualization method that best suits your requirements.

Test and Debug: Thoroughly test your code to ensure it functions correctly. Use sample data or a sandbox environment to validate the accuracy of your analysis and output.

Refine and Enhance: Continuously improve your Stock Market Analyzer by incorporating user feedback and refining your analysis algorithms. Consider adding additional features, such as portfolio management, alert notifications, or integration with other financial tools.

Document and Maintain: Document your code, including comments and user instructions, to make it easier to understand and maintain. Regularly update and maintain your application to keep up with any changes in the data source APIs or to address bugs and improvements.

Remember to refer to relevant documentation and online resources for guidance on specific Java libraries, APIs, and best practices related to stock market analysis. Building a Stock Market Analyzer is a complex project, so taking a systematic and iterative approach will help you create a robust and functional application as mentioned in the previous section.

Examples of comprehensive programming challenges for MSc students learning Java

Examples of some comprehensive programming challenges for MSc students learning Java are presented in Table 2. These programming challenges cover various aspects of Java programming, including data manipulation, algorithms, user interface design, data analysis, and integration with external APIs. They provide MSc students with practical scenarios where they can apply their Java knowledge and skills to solve real-world problems.

Table 2. Examples of some challenges of Java.

Challenges of teaching Java	Explanation	Ref
Implementing a Library Management System	• Design and develop a Java application that allows users to manage a library's inventory. • The system should support functionalities such as adding, updating, and deleting books, managing borrower information, and generating reports.	[18]
Building a Stock Market Analyzer	• Create a Java program that fetches real-time stock market data from an API and performs analysis on the data. • Students can implement features such as calculating stock performance metrics, generating charts and graphs, and providing insights on stock trends.	[19]
Developing a Social Media Analytics Tool	• Design and implement a Java application that analyzes social media data from platforms like Twitter or Facebook. • The program should be able to gather data, perform sentiment analysis, generate user engagement statistics, and visualize the results.	[20]
Creating an Online Shopping Platform	• Build a Java-based e-commerce platform that allows users to browse products, add items to a shopping cart, and complete transactions. • The system should handle user authentication, inventory management, and order processing.	[21]
Designing a Student Management System	• Develop a Java application that manages student information for a university or educational institution. • The system should support functionalities such as enrolling students in courses, tracking grades, managing student records, and generating academic reports.	[22]
Implementing a Travel Booking System	• Create a Java-based system for booking flights, hotels, and other travel services. • The program should handle user registration, search and filtering of available options, reservation management, and payment processing.	[23]
Building a File Encryption and Decryption Tool	• Design a Java application that provides encryption and decryption functionalities for files. • The program should allow users to encrypt sensitive data using encryption algorithms such as AES or RSA, and provide decryption capabilities to restore the original file	[24]

Findings and trends

Based on the reviewed literature, the findings and trends regarding the teaching of Java to MSc students can be summarized as tabulated in Table 3. Taken as a whole, the findings highlight the importance of an interactive and collaborative learning environment supported by tailored IDEs, automated code

analysis tools, comprehensive programming challenges, and progress tracking mechanisms. These trends reflect a shift towards active and practical learning approaches, promoting engagement, critical thinking, and application of Java programming concepts among MSc students.

Table 3. Findings and trends of teaching Java to MSc students.

Findings and trends	Remarks
Interactive Learning Materials	There is a clear emphasis on the importance of interactive learning materials in Java education for MSc students. These materials, such as online tutorials, video lectures, coding exercises, and quizzes, provide hands-on experiences and promote engagement and self-directed learning. Interactive materials help students grasp Java concepts and apply them in practical scenarios.
Integrated Development Environments (IDEs)	Tailored IDEs specifically designed for Java programming play a crucial role in MSc students' learning experiences. These IDEs offer features like code completion, syntax highlighting, debugging tools, and version control integration. They enhance students' productivity and efficiency, enabling them to write, test, and debug Java code effectively.
Collaborative Learning	Collaborative learning is recognized as a valuable approach for teaching Java to MSc students. Real-time code collaboration, peer code reviews, and discussion forums facilitate teamwork, knowledge sharing, and problem-solving skills. Collaborative learning tools promote active participation, encourage peer interaction, and foster a sense of community among students.
Automated Code Analysis and Feedback	The integration of automated code analysis tools and feedback mechanisms is highly beneficial in teaching Java to MSc students. These tools provide instant feedback on code quality, identify errors, and suggest improvements. Immediate feedback allows students to iterate and improve their code, enhancing their understanding of Java programming concepts.
Comprehensive Programming Challenges	Comprehensive programming challenges, such as building applications or systems, are found to be effective in teaching Java to MSc students. These challenges simulate real-world scenarios and require students to apply their Java knowledge and problem-solving skills. They promote critical thinking, creativity, and a deeper understanding of Java programming concepts.
Progress Tracking and Learning Communities	Progress tracking mechanisms and the availability of learning communities and resources contribute to effective Java education for MSc students. Personalized dashboards, progress reports, and performance metrics enable students to monitor their progress and identify areas for improvement. Learning communities provide opportunities for collaboration, support, and continuous learning.

Conclusion

Designing a comprehensive support system for MSc students specializing in Java programming can significantly enhance their educational experience. By incorporating interactive learning materials, an integrated development environment, collaborative learning tools, automated code analysis, comprehensive programming challenges, progress tracking, and access to learning communities, students can acquire a solid foundation in Java programming while fostering collaboration and continuous improvement. As Java continues to evolve, a well-designed support system equips MSc students with the necessary skills and knowledge to excel in their future careers as Java developers. The findings highlight the positive impact of interactive learning materials, tailored IDEs, collaborative learning tools, automated code analysis, comprehensive programming challenges, progress tracking, and learning communities. By implementing such a system, educators can enhance the educational experience and equip MSc students with the necessary skills for successful careers in Java programming. Further research is needed to explore the effectiveness of specific elements within the support system and their impact on student learning outcomes.

References

[1] D. A. Puspita, M. Muchlas, and T. Kuat, “The implementation of teaching factory to improve student interest in entrepreneurship at multimedia competencies,” J. Technol. Humanit., vol. 1, no. 2, pp. 42–50, 2020, [Online]. Available: http://ejournal.jthkkss.com/index.php/jthkkss/article/view/31

[2] Ali Mustofa, Mukhlas Samani, and Soedjarwo, “An Investigation of School Improvement Recognition Towards Madrasah’s Teachers in East Java,” IJORER Int. J. Recent Educ. Res., vol. 2, no. 3, pp. 261–274, May 2021, doi: 10.46245/ijorer.v2i3.112.

[3] N. L. Miftakhurrohmah, M. Masykuri, S. R. D. A. Ariyani, and M. N. Noris, “Effect of Guided Inquiry-Based Excretion System E-Module to Improve Critical Thinking and ICT Literacy Skills for

Students," J. Penelit. Pendidik. IPA, vol. 9, no. 2, pp. 681–689, Feb. 2023, doi: 10.29303/jppipa.v9i2.2036.

[4] A. Sunaryo, S. Hardhienata, E. Suhardi, Junaedi, E. Norman, and F. Salistia, "Improving Teacher Performance Through Strengthening Transformational Leadership, Pedagogical Competence And Organizational Commitment (Empirical Study Using Correlational Approach And Sitorem Analysis In Paud Teachers In Bogor City, West Java Province)-Pa," J. Archaeol. Egypt/Egyptology, vol. 17, no. 6, pp. 1567–214, 2020.

[5] B. Vesin, M. Ivanović, and Z. Budimac, "Learning management system for programming in java," Ann. Univ. Sci. Rolando E{ö}tv{ö}s Nomin. Sect., vol. 31, no. July, pp. 75–92, 2009, [Online]. Available: https://www.researchgate.net/profile/Boban_Vesin/publication/228849378_Learning_Management_System_for_Programming_in_Java/links/00b4953b332db175de000000.pdf

[6] S. Khoirom, M. Sonia, B. Laikhuram, J. Laishram, and D. Singh, "Comparative Analysis of Python and Java for Beginners Cite this paper Comparative Analysis of Python and Java for Beginners," Int. Res. J. Eng. Technol., vol. 07, no. 08, pp. 4384–4407, 2020, [Online]. Available: www.irjet.net

[7] A. Walther, "Viewpoint: From Responsive to Adaptive and Interactive Materials and Materials Systems: A Roadmap," Adv. Mater., vol. 32, no. 20, pp. 1–10, May 2020, doi: 10.1002/adma.201905111.

[8] S. Dhingra, R. B. Madda, A. H. Gandomi, R. Patan, and M. Daneshmand, "Internet of things mobile-air pollution monitoring system (IoT-Mobair)," IEEE Internet Things J., vol. 6, no. 3, pp. 5577–5584, 2019, doi: 10.1109/JIOT.2019.2903821.

[9] A. Chorfi, D. Hedjazi, S. Aouag, and D. Boubiche, "Problem-based collaborative learning groupware to improve computer programming skills," Behav. Inf. Technol., vol. 41, no. 1, pp. 139–158, Jan. 2022, doi: 10.1080/0144929X.2020.1795263.

[10] R. Tufano, L. Pascarella, M. Tufano, D. Poshyvanyk, and G. Bavota, "Towards Automating Code Review Activities," in 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), IEEE, May 2021, pp. 163–174. doi: 10.1109/ICSE43902.2021.00027.

[11] M. Moshirpour, R. Paul, and H. Hemmanti, "DESIGNING A PROGRAMMING BOOTCAMP FOR NON-SOFTWARE ENGINEERS," Proc. Can. Eng. Educ. Assoc., pp. 1–6, Nov. 2019, doi: 10.24908/pceea.vi0.13824.

[12] [12] I. Bafadal, A. Nurabadi, Y. Soepriyanto, and I. Gunawan, "Primary School Principal Performance Measurement," vol. 487, no. Ecpe, pp. 19–23, 2020, doi: 10.2991/assehr.k.201112.004.

[13] W. Yasya, "Rural Empowerment through Education: Case Study of a Learning Community Telecentre in Indonesia," Int. J. Mod. Educ. Comput. Sci., vol. 12, no. 4, pp. 12–26, Aug. 2020, doi: 10.5815/ijmecs.2020.04.02.

[14] Erlangga, Munir, L. Septem Riza, E. Piantari, E. Junaeti, and I. Seanaldi Permana, "Implementation of the Gamification Concept in the Development of a Learning Management System to Improve Students' Cognitive In Basic Programming Subjects Towards a Smart Learning Environment," ADI J. Recent Innov., vol. 5, no. 1, pp. 43–53, May 2023, doi: 10.34306/ajri.v5i1.902.

[15] S. Afrose, S. Rahaman, and D. Yao, "CryptoAPI-Bench: A Comprehensive Benchmark on Java Cryptographic API Misuses," in 2019 IEEE Cybersecurity Development (SecDev), IEEE, Sep. 2019, pp. 49–61. doi: 10.1109/SecDev.2019.00017.

[16] A. S. Bedi, R. Sparrow, and L. Tasciotti, "The impact of a household biogas programme on energy use and expenditure in East Java," Energy Econ., vol. 68, pp. 66–76, Oct. 2017, doi: 10.1016/j.eneco.2017.09.006.

[17] F. S. Gazijahani and J. Salehi, "Reliability constrained two-stage optimization of multiple renewable-based microgrids incorporating critical energy peak pricing demand response program using robust optimization approach," Energy, vol. 161, pp. 999–1015, Oct. 2018, doi: 10.1016/j.energy.2018.07.191.

[18] Katsanevakis, Bennett, and Stewart, "Economic, Social and Environmental Dimensions of PHEV in the Smart Grid," in Vehicle-to-Grid: Linking electric vehicles to the smart grid, Institution of Engineering and Technology, 2015, pp. 223–252. doi: 10.1049/PBPO079E_ch7.

[19] B. R. Kartawinata, M. Fakhri, M. Pradana, N. F. Hanifan, and A. Akbar, "The Role Of Financial Self-Efficacy: Mediating Effects Of Financial Literacy & Financial Inclusion Of Students In West Java, Indonesia," J. Manag. Inf. Decis. Sci., vol. 24, no. Special Issue 2, pp. 1–9, 2021.

[20] A. Rahman and A. Hori, “A Journal of Vytautas Magnus University Quality improvement management of private islamic religious education through empowerment of educational personnel (Study at the Operators Forum of Islamic Religious Colleges in Kopertais Region II West Java),” Balt. J. Law Polit., vol. 16, no. 2, pp. 251–268, 2023, doi: 10.2478/bjlp-2023-0000020.

[21] I. M.said, Heri Cahyo Bagus Setiawan, Nuzulul Fatimah, and Tatag Herbayu L, “Change Business Model of Islamic Religious College Business in East Java By Building Integrated Online Policy and Technology Systems During the Covid-19 Pandemic Period,” J. Islam. Econ. Perspect., vol. 2, no. 1, pp. 66–83, 2020, doi: 10.35719/jiep.v2i1.34.

[22] C. Sunaengsih, M. Anggarani, M. Amalia, S. Nurfatmala, and S. D. Naelin, “Principal Leadership in the Implementation of Effective School Management,” Mimb. Sekol. Dasar, vol. 6, no. 1, p. 79, Apr. 2019, doi: 10.17509/mimbar-sd.v6i1.15200.

[23] S. T. Safitri, C. Wiguna, and I. Y. Wibowo, “Comparison of factors that affect behavior intention on travel booking applications,” 2023, p. 100009. doi: 10.1063/5.0113888.

[24] M. Zhang, Y. Bai, S. Yuan, N. Tian, and J. Wang, “Design and Implementation of File Multi-Cloud Storage System Based on Android,” in 2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS), IEEE, Oct. 2020, pp. 212–215. doi: 10.1109/ICSESS49938.2020.9237695.