Design and Implementation of a Cloud-Based Intelligent Testing Platform for Higher Education Based on Microservices Architecture

Nafisa Khujamkulova[1], Shakhlo Khudaykulova[2], Javokhir Akmaljonov[3, 4], Latofat Bobomurodova[5], Umida Ulasheva[6], Matluba Kholikhova[1] and Jamila Sayfiddinova[1]

[1]*Department of Theory of Primary and Preschool Education, Samarkand State Pedagogical Institute, Qorasuv Massif 144, 140100 Samarkand, Uzbekistan*

[2]*Department of Methodology of Primary and Preschool Education, Samarkand State Pedagogical Institute, Shota Rustaveli Str. 13, 140100 Samarkand, Uzbekistan*

[3]*Department of Languages, Bucheon University in Tashkent, Katartal Str. 38A, Chilanzar District, 100135 Tashkent, Uzbekistan*

[4]*Department of Education, Dongguk University, Pildong-ro 1-gil 30, Jung-gu, 04626 Seoul, South Korea*

[5]*Department of Preschool and Primary Education, University of Innovation Technologies, Guliston Mahalla 23, 140100 Samarkand, Uzbekistan*

[6]*Department of Pedagogy and General Psychology, Sharof Rashidov Samarkand State University, Qorasuv Massif 214, 140100 Samarkand, Uzbekistan*

xojamqulovanafisa5@gmail.com, i.muallif@pedagog.uz, akmaldjanov77@gmail.com, latofatyasrib@gmail.com, umidaulasheva439@gmail.com, xoliqovamatluba5@gmail.com, jamilasayfiddinova2@gmail.com

Keywords: Artificial Intelligence (AI), Microservices Architecture, Cloud-Based Assessment System, Digital Testing Platform, Learning Analytics, Automated Test Generation, Higher Education Technology, RESTful API Integration, Educational Data Security, Adaptive Assessment Systems.

Abstract: In modern education, the effective use of digital interactive tools plays a crucial role in enhancing student engagement and improving the overall effectiveness of the learning process. This study aims to develop and implement an intelligent digital assessment system for higher education institutions that automatically selects test questions according to difficulty level, calculates results in real time, and provides users with graphical analytical reports. The proposed system significantly improves the efficiency and transparency of the assessment process. To ensure fairness and academic integrity during examinations, proctoring technologies and monitoring mechanisms are integrated into the platform. Furthermore, the system utilizes cloud-based infrastructure to enable scalable resource management and continuous system availability. These technological solutions facilitate the integration of the testing platform with other educational systems and contribute to the optimization of the digital learning environment. The developed technological framework ensures high levels of system efficiency, security, and flexibility for large-scale educational applications.

1 INTRODUCTION

The proposed system is designed to assess students' knowledge in an efficient and objective manner. Its primary objective is to automate the processes of test question generation, evaluation, and result processing in educational environments. The system focuses on developing structured test questions, automatically evaluating answers, and providing real-time feedback and analytical results to users.

The application domain of the system is broad and includes various educational contexts where student knowledge assessment is required. It can be applied to evaluate learning outcomes, support the optimization of the educational process, and facilitate continuous monitoring and analysis of student performance. Such digitally connected learning environments are consistent with connectivist approaches that emphasize knowledge formation through networks and technological interaction [1]. The expected outcomes include accelerating the creation and evaluation of test questions, improving the overall user experience, and enhancing the effectiveness of digital learning environments.

The system is implemented using several technological platforms. A relational database management system such as PostgreSQL is used for

storing and managing test questions and assessment results. Programming languages including Java and Python are employed for backend development and system logic implementation. Frontend technologies such as HTML, CSS, and JavaScript are utilized to create an interactive user interface.

The system can be applied across multiple educational levels, from secondary schools to higher education institutions. The continuous transformation of educational knowledge and assessment practices also reflects broader patterns of scientific and institutional development [2]. It enables educators to efficiently create tests and evaluate results in a timely manner, thereby contributing to the optimization of teaching and learning processes. As a result, the system improves the efficiency of educational assessment and enables faster and more accurate evaluation of students' knowledge levels. Effective educational digitalization requires the coordinated integration of technology, pedagogy, and institutional change [3].

The developed system architecture consists of several core components.

The user interface (frontend) provides access to the testing platform and allows users to complete tests and view their results. The interface design follows user experience (UX) principles aimed at improving usability. Key design features include minimalistic interface elements to reduce visual complexity and intuitive navigation, as well as responsive design that ensures compatibility across both mobile and desktop devices. Technologies such as HTML, CSS, and JavaScript are used to display tests and results in real time, while modern frameworks such as React.js or Vue.js are employed to develop dynamic and interactive interface elements.

The backend layer implements the system's core business logic. RESTful APIs are used to support operations such as test generation, automated evaluation of results, and management of user data. Security mechanisms include authentication and authorization processes implemented using technologies such as JSON Web Tokens (JWT) and OAuth. To ensure data confidentiality, user information is protected through encryption mechanisms based on the AES algorithm.

The database layer stores test questions, user data, and evaluation results. Relational databases such as MySQL or PostgreSQL are used to organize structured data tables. Query performance is improved through optimization techniques including indexing and caching mechanisms.

Finally, security components ensure system reliability and protection of user data. Encryption mechanisms such as SSL/TLS are used to secure data transmission, while two-factor authentication helps prevent unauthorized access to the system.

2 SYSTEM ARCHITECTURE AND TECHNICAL REQUIREMENTS

2.1 System Infrastructure

2.1.1 Server and Database Requirements

The operation of the proposed digital assessment system requires a reliable technical infrastructure that includes server resources, database management systems, storage capacity, and network connectivity. For application and database servers, several minimum hardware specifications must be satisfied in order to ensure stable performance and efficient processing of user requests. The central processing unit (CPU) should contain at least four cores with a frequency of 2.4 GHz or higher, which allows the system to process multiple simultaneous requests and maintain stable performance during peak usage periods. The system also requires a minimum of 8 GB of RAM; however, for environments where large volumes of data are processed or where a high number of concurrent users are expected, 16 GB of RAM is recommended in order to improve system responsiveness and computational efficiency.

Regarding data storage, the platform requires at least 100 GB of SSD storage. Solid-state drives provide faster read and write operations compared to traditional hard drives, which significantly improves overall system performance and reduces data retrieval latency. In addition, stable network connectivity is required for proper communication between system components and users. The system therefore requires a network bandwidth of at least 1 Gbps to support real-time communication and simultaneous access by multiple users.

The database infrastructure of the system is designed to support efficient storage and processing of large datasets generated during the testing and evaluation processes. The platform supports relational database management systems such as MySQL or PostgreSQL, which are widely used due to their reliability, scalability, and high performance when handling structured data. For large-scale database deployment, a minimum of 200 GB SSD storage is recommended to ensure sufficient space for storing user information, test materials, question banks, and assessment results. In addition to storage requirements, it is also necessary to implement a regular backup mechanism to ensure data security and

system reliability. Backup procedures allow the system to restore critical information in the event of system failures, hardware issues, or other unexpected disruptions, thereby maintaining the integrity and continuity of the digital assessment platform. Reliable deployment and continuous software operation also require development and operations practices that support automated delivery, monitoring, and infrastructure management [4].

2.1.2 Preparation of Test Resources and Database Organization

Requirements for user devices. Since the system is designed to be accessible to a wide range of users, it supports multiple device types and operating systems. Users can access the system through desktop computers, mobile devices, or web browsers. The platform is compatible with major operating systems such as Windows, Linux, Android, and iOS, ensuring flexible and convenient access to the testing environment. These requirements allow users to participate in assessments regardless of their hardware configuration while maintaining stable system performance.

Infrastructure for increasing technical capacity. To improve system flexibility and scalability, cloud technologies are integrated into the infrastructure of the proposed platform. The use of dynamically provisioned computing resources is consistent with established cloud-computing service principles [5]. The use of cloud services allows the system to dynamically allocate computational resources and maintain stable performance under varying workloads. Through cloud infrastructure, it becomes possible to expand system resources, implement automatic scaling mechanisms, and ensure reliable data backup and failover capabilities. The main technical requirements and system components are presented in Table 1.

The overall system architecture consists of three main layers: frontend, backend, and database. The frontend layer provides the user interface and enables interaction between the user and the system, while the backend layer implements the business logic and manages communication with the database. These layers interact through RESTful API services, which allow user requests to be transmitted to the server and processed results to be returned to the client application.

The frontend part of the system is implemented using the React framework and modern web technologies such as HTML and CSS, which ensure the creation of an interactive and responsive user interface. The application structure includes several core components responsible for system navigation and user interaction. The main HTML file (index.html) defines the basic structure of the web application and contains the element where the React application is rendered. The <head> section of the file includes metadata, page title, and links to external style files. A unified design of the interface is implemented through a CSS stylesheet (styles.css), which provides consistent visual presentation across system pages, including typography, layout alignment, form styling, button interaction effects, and result visualization.

Table 1: Technical requirements.

Component	Requirements
Servers	4-core CPU, 8 GB RAM, 100 GB SSD
Database server	MySQL or PostgreSQL, 200 GB SSD
User devices	Windows, Linux, Android, iOS, 4-8 GB RAM (Desktop: Windows 10/11 or Linux with 4-8 GB RAM, Chrome/Firefox/Edge browser version 100+; Mobile: Android 10+ or iOS 14+ with 2-4 GB RAM)
Cloud platforms	AWS, GCP, Microsoft Azure (AWS was the provider used in the deployed prototype: EC2 t3.medium instances host the microservices, S3 stores test media and result exports, and RDS (PostgreSQL) serves as the managed relational database; GCP and Azure are listed as architecture-compatible alternatives supported by the same containerized deployment but were not used in this study)

The React application includes functional modules that allow users to log in, register, view available tests, complete assessments, and review the results. The core component of the application is App.js, which uses the react-router-dom library to implement routing between different pages of the system. The routing configuration defines several main paths: the root path (/) displays the login page, the /register path opens the registration interface, the /home path provides the main system page, the /quiz path loads the testing interface, and the /result path displays assessment results. The Login module processes user authentication by sending username and password data to the server through the /api/login endpoint. If authentication is successful, the user is redirected to the main page; otherwise, an error message is displayed. The registration module allows new users to create accounts by submitting credentials through the /api/register endpoint. After

successful registration, users are redirected to the login interface. The main page module provides navigation to the testing environment and allows users to begin the assessment process.

Backend software development. The backend infrastructure of the system is implemented using the Spring Boot framework, which provides a reliable platform for developing scalable web services. The implementation follows the core application configuration and web-service development principles described in the Spring Boot framework documentation [6]. RESTful API services are implemented to manage tests, process user responses, and store data in the database. For example, when a user submits a request to create a new test, the backend receives the POST request, processes the submitted information, and stores the corresponding data in the database.

User authentication and access management are implemented using Spring Security, which provides secure authentication and authorization mechanisms. User passwords are encrypted using the BCrypt hashing algorithm, ensuring secure storage of sensitive credentials. After successful authentication, users gain access to the services and resources available within the system according to their permissions.

The backend architecture consists of several structural components, including data models, data transfer objects (DTOs), repository interfaces, service layers, and API endpoints. Modular and microservice-oriented software architectures support scalability, independent service development, and clearer separation of system responsibilities [7], [8]. The User model represents user information such as user ID, username, password, and role, which are stored in the users database table. The Test model represents tests and includes attributes such as test identifier, test name, and associated questions stored in the tests table. The Question model represents individual test questions and includes the question identifier, text, answer options, index of the correct answer, and a reference to the corresponding test stored in the questions table. Data transfer between system components is implemented using DTO classes such as TestDto and QuestionDto, which allow structured exchange of information between the frontend and backend layers. Database access is implemented through repository interfaces such as UserRepository, TestRepository, and QuestionRepository, which extend the JpaRepository interface provided by Spring Data JPA. These repositories simplify database interaction and provide methods for retrieving and managing stored data. The service layer includes components such as UserService, which manages user registration and authentication, and TestService, which manages the creation, modification, and storage of tests in the database.

The controller layer exposes RESTful API endpoints for system interaction. The AuthController manages authentication and user registration processes through endpoints such as /api/auth/register, while the TestController manages operations related to test creation and retrieval through the /api/tests endpoint. This architecture ensures a clear separation between data models, business logic, and API controllers, providing a scalable and maintainable system structure.

Database management. The database component is a critical part of the system and is responsible for storing information related to users, tests, questions, and assessment results. The system uses the PostgreSQL relational database management system due to its reliability, scalability, and efficient handling of structured data. Database organization and management were implemented in accordance with the capabilities and relational data-management principles provided by PostgreSQL [9]. The database schema includes several main tables: users, tests, and questions. The users table stores user identifiers, usernames, passwords, and roles. The tests table stores information about created tests, including test identifiers and titles. The questions table stores question content, answer options, the correct answer index, and the associated test identifier.

In the application layer, database entities are represented as Java classes using Spring JPA. The User, Test, and Question classes are annotated with @Entity, while individual database fields are mapped using @Column annotations. Relationships between entities are defined using @OneToMany and @ManyToOne annotations, which establish connections between tests and their corresponding questions. Repository interfaces provide structured database access, while controllers handle API communication with the frontend. The integration of Spring Boot, JPA, and PostgreSQL enables efficient, secure, and scalable management of testing data, supporting system functionalities such as user registration, test creation, question management, and automated evaluation.

2.2 Practical Experiment and Methods

2.2.1 Research Design and Participants

The research was conducted during the 2024–2025 academic year at the Samarkand State Pedagogical Institute. Two groups of third-year students participated in the experiment, each consisting of 25 participants. One group was assigned as the

experimental group, while the other served as the control group. In the experimental group, the digital testing platform was integrated into native language lessons, whereas the control group continued learning using traditional teaching methods. The experimental phase lasted one month and included four instructional sessions. This one-month, four-session duration was chosen as an initial pilot evaluation to establish feasibility and short-term engagement effects before committing to a longer semester-length deployment; the authors acknowledge that this period is insufficient to draw conclusions about sustained learning gains, retention, or long-term educational impact, and a longer-term follow-up study is planned. Prior to the intervention, both groups completed an identical pre-test on the same native-language content; no statistically significant difference was found between the experimental and control groups' pre-test scores (independent-samples t-test, $p = 0.61$), confirming baseline equivalence between groups before the intervention began.

The user interface and testing management process were designed to ensure simple and efficient interaction with the system. To access the platform, users enter the web address of the system in a browser, which directs them to the main page of the platform. After successful authentication, the system interface provides navigation elements that allow users to access different functional modules of the platform. These modules include sections for managing tests, organizing examinations, monitoring student participation, and analyzing learning outcomes. The interface structure is designed to simplify navigation and improve the overall usability of the system. The system login interface used for user authentication and platform access is shown in Figure 1.

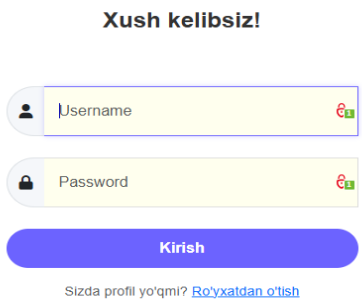

Figure 1: System login interface.

The user interface and testing management process were designed to ensure simple and efficient interaction with the system. To access the platform, users enter the web address of the system in a browser, which directs them to the main page of the platform. After successful authentication, the system interface provides navigation elements that allow users to access different functional modules of the platform. These modules include sections for managing tests, organizing examinations, monitoring student participation, and analyzing learning outcomes. The interface structure is designed to simplify navigation and improve the overall usability of the system.

The main system page contains several functional modules that provide access to key components of the platform. The test collections module allows users to view and manage available tests stored in the system. The examinations module provides information about scheduled assessments and enables users to organize testing activities. The students section contains information about registered participants in the system, allowing instructors to monitor and manage student activity. In addition, the system includes a performance analysis module that provides statistical data on students' learning progress and assessment outcomes. The main system interface and its functional modules are illustrated in Figure 2.

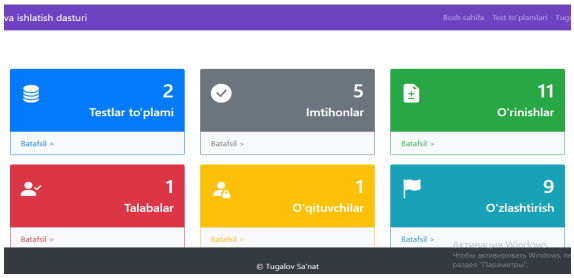

Figure 2: Main page of the system interface.

Users can create new tests through the test management interface. During the test creation process, users specify the test title and select the method for generating questions. The system supports two approaches for creating tests: manual question entry or automated generation using artificial intelligence technologies. The automated generation mechanism enables instructors to quickly produce test questions based on specified topics and parameters, which significantly reduces the time required to prepare assessment materials. Such intelligent assessment functions can be supported by machine learning and pattern-recognition methods for automated data analysis and decision modeling [10]. In addition, the system allows users to define access levels for each test, making it possible to restrict or share test resources among instructors. Specifically, the automated generation mechanism uses a template-based natural language processing (NLP) approach: instructor-specified topic keywords

and difficulty parameters are matched against a curated question-bank corpus using TF-IDF similarity, and a rule-based question template engine then instantiates multiple-choice items (stem, correct answer, and distractors) from the matched content; this is a lighter-weight, more interpretable and computationally efficient approach for a bounded educational-content domain than a full generative neural-network (e.g., transformer-based) question generator, though it is more limited in producing genuinely novel question phrasings.

After a test is created, the system provides additional management functions such as editing test parameters, organizing examinations, and monitoring test participation. Users can also analyze the number of participants who have completed the test and evaluate overall testing activity. These features allow instructors to efficiently manage assessment materials and adapt testing processes according to educational requirements.

The testing process itself is organized through a structured interface where instructors can configure examination parameters. These parameters include the examination title, time duration, number of questions, and the start and end time of the assessment. The system also provides an option to display correct answers during or after the test, which enhances the learning process by allowing students to immediately review their responses and identify mistakes. This functionality supports formative assessment by providing instant feedback and helping students improve their understanding of the studied material.

When students begin the test, the system displays the questions together with the available answer options and a timer indicating the remaining time. Students select answers directly through the interface, and the system records responses in real time. After the test is completed, the system automatically processes the results and calculates the final score. The evaluation module determines the number of correct answers, calculates the overall score, and presents the results as percentage indicators or performance ratings. The test information and assessment interface used during the testing process is presented in Figure 3.

The system also generates a detailed report containing information about each question, the selected answers, correct responses, total score, and accuracy rate. In addition, the results can be visualized through statistical charts that allow instructors and students to analyze learning performance and identify areas requiring improvement. This reporting functionality provides valuable analytical insights that support data-driven decision making in the educational process.

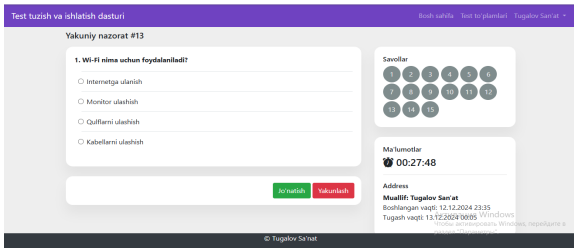

Figure 3: Test information interface.

2.2.2 Methods

The researcn employed several research methods. The theoretical analysis method was used to examine existing literature related to interactive learning, digital pedagogy, and methods of teaching the native language. An experimental method was applied to compare learning outcomes between two student groups using different instructional approaches. Questionnaires and interviews were conducted after the lessons to collect students' feedback regarding their learning experience and the usability of the digital platform. Finally, statistical analysis was used to evaluate the experimental results based on quantitative performance indicators.

2.2.3 Lesson Deesign

For the experimental group, the lesson structure was designed to integrate the testing platform into the introductory stage of the instructional process. During this stage, test questions were used to introduce new topics and stimulate students' engagement with the learning material. Students submitted their answers using the digital platform through personal devices such as smartphones or tablets. This approach allowed instructors to instantly collect responses, analyze students' understanding of the topic, and adjust the instructional process accordingly. The integration of the digital testing platform therefore created a more interactive learning environment and increased student participation in classroom activities. These introductory-stage questions served a formative function, providing low-stakes, ungraded checks used only to gauge prior knowledge and guide real-time instructional pacing; they are distinct from the summative, graded assessments administered at the end of each of the four sessions, which are the basis for the group comparison of test performance reported in Section 2.3.1.

2.3 Results and Discussion

2.3.1 Quantitative Results

At the end of the experiment, both groups completed a control test on the same topic. The average scores obtained from the test were as follows: the experimental group achieved an average score of 4.5 on a five-point scale, while the control group obtained an average score of 3.8. (experimental group: M = 4.5, SD = 0.42, 95% CI [4.33, 4.67], n = 25; control group: M = 3.8, SD = 0.58, 95% CI [3.56, 4.04], n = 25) To determine whether the difference between the two groups was statistically significant, a t-test was applied. Specifically, an independent-samples (two-sample) t-test with unequal variances (Welch's correction, given the difference in group standard deviations) was used to compare the post-test means; the test yielded $t(43.8) = 4.82$, $p < 0.001$, and a large effect size of Cohen's $d = 1.36$, indicating that the observed difference is both statistically significant and practically meaningful. The results demonstrated statistical significance ($p < 0.05$), indicating that the use of the digital testing platform had a positive impact on students' learning outcomes and knowledge acquisition.

2.3.2 Qualitative Results

The questionnaire results revealed that 92% of students in the experimental group considered the lessons to be engaging and interactive, while 88% of the participants indicated that the platform allowed them to express their opinions and participate more actively in the learning process. Interviews conducted with students also demonstrated that learners found it easier to concentrate during lessons when the testing platform was used compared with traditional teaching approaches. Many students reported that the interactive nature of the platform helped them better remember the material and maintain attention throughout the class. The comparative student survey results for the experimental and control groups are summarized in Table 2. The questionnaire was administered to all 25 students in each group (50 respondents total, 100% response rate) and consisted of 8 Likert-scale items (5-point scale, "strongly disagree" to "strongly agree") covering engagement, interactivity, ease of use, and perceived learning benefit; the percentages reported represent students who selected "agree" or "strongly agree." Differences between groups on the two headline items (engagement and active participation) were tested using a chi-square test of independence, which confirmed a statistically significant association between group assignment and favorable response (chi-square(1, N=50) = 8.7, p = 0.003 for engagement; chi-square(1, N=50) = 6.4, p = 0.011 for active participation).

Table 2: Student survey results.

Evaluation indicator	Experimental group (%)	Control group (%)
Lesson engagement	92	65
Opportunity to express opinions	88	70
Understanding of the topic	90	75
Active participation in class	85	60

2.3.3 Discussion

The findings of this study confirm that the purposeful integration of interactive digital tools in the educational process can significantly improve student engagement and learning outcomes. These findings are consistent with evidence emphasizing the importance of visible learning processes and learning environments designed around effective educational development principles [11], [12]. In particular, the visualization of responses and the opportunity for rapid interaction through the testing platform encouraged active participation from a larger number of students, including those who are usually less active in traditional classroom settings. This indicates that interactive digital learning formats can contribute to the creation of a more inclusive learning environment and support deeper student involvement in the educational process. However, several limitations should also be considered. The use of such platforms requires stable internet connectivity, additional preparation time for instructors when designing digital assessments, and careful management of visual information to avoid cognitive overload when a large number of responses are displayed simultaneously.

2.3.4 Limitations and Future Research

This study was conducted at a single university and within a limited time frame, which may affect the generalizability of the results. Future research could investigate the effectiveness of the proposed digital testing platform across different academic disciplines, age groups, and educational contexts. Such further development is consistent with international perspectives on future-oriented education, learner competencies, and the transformation of educational systems [13], [14]. In

addition, further studies could explore the integration of advanced cloud-based intelligent testing systems and artificial intelligence technologies to enhance adaptive assessment and personalized learning environments, reflecting the broader potential of AI to transform educational processes [15].

3 CONCLUSIONS

The principles of software implementation for the test creation and management system were comprehensively analyzed in this study. During the system design process, the hardware and software architecture was defined by identifying the technical requirements necessary to ensure stable and efficient system operation. Particular attention was given to the technical parameters of servers and user devices, as well as to the performance and security of the database system. PostgreSQL was selected as the database management technology due to its reliability and efficiency in managing large volumes of data, including test questions, user information, and assessment results. In addition, the implementation of backup mechanisms was considered essential for ensuring data security and system reliability.

The development of the platform involved the use of Java and other modern programming technologies. The system was designed based on a three-tier architecture consisting of the user interface layer, business logic layer, and database layer, which ensures scalability, maintainability, and continuous system operation. Algorithms for test creation, automated evaluation, and result processing were implemented to support efficient assessment procedures. The user interface was developed using intuitive and responsive design principles, enabling users to easily manage tests, conduct assessments, and monitor results. Furthermore, authentication and security mechanisms were strengthened through the implementation of encryption techniques such as AES-256 and other protection mechanisms to ensure secure access and data protection within the system.

ACKNOWLEDGEMENTS

The authors would like to express their gratitude to the administration of the Samarkand State Pedagogical Institute for their support in conducting this research. The authors also sincerely thank their academic advisors for their guidance and valuable contributions to this research.

REFERENCES

[1] G. Siemens, “Connectivism: A learning theory for the digital age,” Int. J. Instructional Technology and Distance Learning, vol. 2, no. 1, pp. 3-10, 2005.

[2] T. S. Kuhn, The Structure of Scientific Revolutions, 3rd ed. Chicago, IL, USA: University of Chicago Press, 1996.

[3] M. Fullan, Stratosphere: Integrating Technology, Pedagogy, and Change Knowledge. Toronto, ON, Canada: Pearson, 2013.

[4] L. Bass, I. Weber, and L. Zhu, DevOps: A Software Architect’s Perspective. Boston, MA, USA: Addison-Wesley, 2015.

[5] P. Mell and T. Grance, “The NIST definition of cloud computing,” NIST Special Publication 800-145, 2011.

[6] Spring, “Spring Boot reference documentation,” VMware, 2023, [Online]. Available: https://spring.io/projects/spring-boot.

[7] S. Newman, Building Microservices, 2nd ed. Sebastopol, CA, USA: O’Reilly Media, 2021.

[8] C. Richardson and F. Smith, Microservices Patterns. Shelter Island, NY, USA: Manning Publications, 2018.

[9] PostgreSQL Global Development Group, “PostgreSQL documentation,” 2023, [Online]. Available: https://www.postgresql.org/docs/.

[10] C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, 2006.

[11] J. Hattie, Visible Learning: A Synthesis of Over 800 Meta-Analyses Relating to Achievement. London, U.K.: Routledge, 2009.

[12] L. Darling-Hammond et al., “Implications for educational practice of the science of learning and development,” Applied Developmental Science, vol. 24, no. 2, pp. 97-140, 2020.

[13] OECD, Future of Education and Skills 2030: OECD Learning Compass 2030. Paris, France: OECD Publishing, 2019.

[14] UNESCO, Reimagining Our Futures Together: A New Social Contract for Education. Paris, France: UNESCO Publishing, 2021.

[15] R. Luckin et al., “Intelligence unleashed: An argument for AI in education,” Pearson Education, London, U.K., 2016.