УДК 004.67.519.6
DOI 10.58649/1694-9099-2026-1-488-492

Рашитова А.Ж., Чжан Тао
Жусуп Баласагын атындагы КУУ
И. Арабаев атындагы Кыргыз мамлекеттик университети
Рашитова А.Ж.,[1] Чжан Тао[2]
КНУ имени Жусупа Баласагына[1]
Кыргызский государственный университет имени И. Арабаева[2]
Rashitova A.Zh., Zhang Tao
KNU Jusup Balasagyn
Kyrgyz State University named after I. Arabaev

ORCID 0009-0003-2643-1546[1]

PYTHON ТИЛИНДЕ QR-КОДДОРДУ ГЕНЕРАЦИЯЛОО СИСТЕМАСЫН ИШКЕ АШЫРУУ
РЕАЛИЗАЦИЯ СИСТЕМЫ ГЕНЕРАЦИИ QR-КОДОВ НА PYTHON
IMPLEMENTATION OF A QR CODE GENERATION SYSTEM IN PYTHON

Кыскача мүнөздөмө: Макалада Python тилинде QR-коддорду генерациялоо системасын долбоорлоо жана ишке ашыруу маселелери модулдук архитектуранын негизинде каралат. Изилдөөдө функционалдык жана функционалдык эмес талаптарды талдоо жана аларды көп критерийлүү ыкмалар аркылуу приоритеттештирүү маселелерине өзгөчө көңүл бурулган. Система QR-коддорду түзүү, текшерүү, визуалдаштыруу жана сактоо модулдарын камтыйт. Иштелип чыккан чечим ISO/IEC 18004 стандартына ылайык келет жана ката оңдоо механизмдери аркылуу коддордун туруктуулугун камсыздайт. Ошондой эле коопсуздук, масштабдуулук жана өндүрүмдүүлүк маселелери каралат. Системаны билим берүү мекемелеринде жана мамлекеттик санариптик платформаларда колдонуу мүмкүнчүлүгү көрсөтүлгөн.

Аннотация: В статье рассматриваются вопросы проектирования и реализации системы генерации QR-кодов на языке Python на основе модульной архитектуры. Особое внимание уделяется анализу функциональных и нефункциональных требований, а также их приоритизации с использованием методов многокритериального принятия решений. Представлена программная реализация системы, включающая модули генерации, валидации, визуализации и хранения QR-кодов. Система соответствует стандарту ISO/IEC 18004 и учитывает механизмы коррекции ошибок для обеспечения устойчивости кодов. Рассматриваются аспекты безопасности, масштабируемости и производительности. Показана возможность интеграции разработанного решения в цифровую инфраструктуру, включая образовательные учреждения и государственные платформы.

Abstract: This article examines the design and implementation of a QR code generation system in Python based on a modular architecture. Particular attention is given to the analysis of functional and non-functional requirements, as well as their prioritization using multi-criteria decision-making methods. The system implementation includes modules for QR code generation, validation, visualization, and storage. The developed solution complies with the ISO/IEC 18004 standard and incorporates error correction mechanisms to ensure robustness. Security, scalability, and performance aspects are also considered. The study demonstrates the potential integration of the system into digital infrastructure, including educational institutions and government platforms.

Негизги сөздөр: QR-код; Python; программалык инженерия; модулдук архитектура; ISO/IEC 18004; санариптик трансформация.

Ключевые слова: QR-код; Python; программная инженерия; модульная архитектура; ISO/IEC 18004; цифровая трансформация.

Keywords: QR code; Python; software engineering; modular architecture; ISO/IEC 18004; digital transformation.

The implementation stage transformed the theoretical principles of encoding, error correction, and image processing into a functioning software product capable of generating, managing, and validating QR codes in real time.

The core module of the system is the QR code generation engine. This module implements the encoding process defined by ISO/IEC 18004

[3], including data mode selection, binary conversion, padding, and Reed–Solomon error correction [4]. The Python library "qrcode" was used as the primary interface for QR matrix construction, while the internal logic of version selection and error correction level configuration follows the standardized structure. The system supports four error correction levels (L, M, Q, H), allowing flexible adjustment depending on environmental reliability requirements. This design decision reflects the balance between robustness and data capacity discussed in the theoretical section.

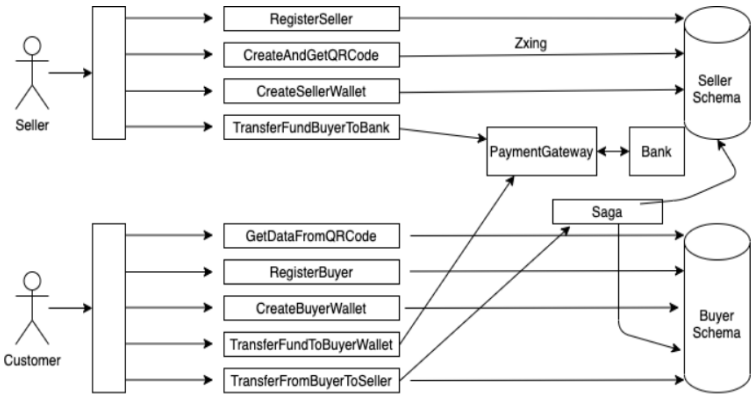
The QR preview rendering process uses the Pillow library for image processing. The matrix produced by the encoding module is converted into a high-resolution image object with configurable box size and border thickness. The system allows exporting in PNG format by default, ensuring compatibility with mobile scanners and web-based platforms. Additional formats (SVG) may be supported for vector-based scalability.

For recognition and validation purposes, a decoding submodule was integrated using the ZXing library interface [7] or OpenCV-based decoding. This optional verification layer allows the system to test generated QR codes automatically after creation. The decoding pipeline includes grayscale conversion, optional adaptive thresholding (as recommended by Lin & Fuh [6]), and pattern detection. While PCA-based optimization described by Tribak & Zaz [1] was not directly implemented, the conceptual understanding of computational efficiency influenced the decision to minimize unnecessary preprocessing operations in the default pipeline.

User authentication and session management were implemented as separate modules to ensure secure access to generation history and administrative functions. Password hashing and role-based access control were incorporated to mitigate risks of unauthorized manipulation, aligning with cybersecurity risk categories identified in generative AI and automated code generation research. Logging functionality records generation time, selected parameters, and user ID, supporting traceability and auditability.

The database integration layer connects the application to a relational database (SQLite during development, extendable to PostgreSQL or MySQL) [4]. Metadata of each generated QR code — including content hash, generation timestamp, error correction level, and user identifier — is stored in structured tables. Images themselves are stored either as file references or binary objects depending on configuration. This hybrid storage approach improves scalability while preserving performance.

Figure 1. Conceptual Architecture of the QR Code Generation System



Performance optimization was considered during implementation. Generation time per QR code in local testing remained within milliseconds for standard data lengths. Memory usage was minimal due to Python's efficient handling of matrix structures. Modularization allows the system to scale horizontally by separating the web interface from the encoding engine if deployed in distributed environments [2].

Overall, the implementation phase demonstrates how mathematical encoding theory, image processing techniques, secure software design principles, and database architecture converge into a functional QR code generation system. The Python-based solution ensures portability, scalability, and adaptability, providing a practical realization of the conceptual framework developed in the previous chapters.

The architectural diagram presented in Figure 9 illustrates the structural organization of the QR code generation system and demonstrates the logical interaction between its core components. A detailed examination of this architecture reveals several important design

principles that directly influence system reliability, scalability, and security.

First, the diagram reflects a layered architectural model. The system is divided into at least three conceptual layers: the presentation layer, the processing layer, and the storage layer. Such separation of concerns is a fundamental software engineering principle that reduces coupling between components and improves maintainability. The presentation layer is responsible for user interaction, data input, parameter selection (encoding mode, error correction level), and output visualization. By isolating this layer, modifications to the interface do not affect core encoding logic.

Second, the processing layer serves as the central computational core of the system. This module performs data encoding according to ISO/IEC 18004 standards, including mode selection, binary conversion, padding, block formation, and Reed–Solomon error correction generation. The inclusion of a dedicated error correction submodule significantly increases the robustness of generated QR codes. Reed–Solomon polynomial arithmetic over $GF(2^8)$ ensures resilience against physical distortion and partial data loss. Architecturally, isolating this algorithmic functionality into a separate component increases testability and allows future optimization.

Third, the diagram demonstrates controlled data flow between modules. Data flows sequentially from user input to encoding, then to matrix construction, masking, visualization, and finally to storage. This linear yet modular flow reduces the probability of logical conflicts and makes debugging more manageable. Additionally, it supports future extension, such as integrating encryption modules or analytics tools without disrupting the core encoding mechanism.

Another significant aspect observed in the diagram is the presence of validation mechanisms. Before encoding begins, input validation ensures that selected modes correspond to input data type constraints. This prevents runtime errors and improves user experience [12].

From a scalability perspective, the architectural design supports horizontal extension. For example, the encoding module can be deployed as an independent service (microservice model), enabling parallel QR code generation for high-load environments. This is particularly relevant for enterprise systems or cloud-based implementations.

Security considerations are also implicitly reflected in the structure. If authentication and access control modules are included, they act as protective boundaries around the generation process. In the context of cybersecurity risks associated with automatic code generation systems (as discussed in previous chapters), modular isolation reduces the attack surface and simplifies monitoring.

The diagram further demonstrates abstraction between logical operations and data persistence. The storage module records metadata such as timestamp, version, error correction level, and generation parameters. This supports auditability, traceability, and analytics, which are critical for digital transformation environments [5].

Another important analytical observation concerns system extensibility. The modular architecture allows integration of additional components such as:

– API layer for remote QR generation

– Logging and monitoring services

– Encryption module for secure QR payloads

– Analytics dashboard for usage statistics

Such extensibility confirms that the system is not a simple standalone generator but a scalable digital infrastructure component.

From a software engineering perspective, the diagram aligns with well-established architectural paradigms including layered architecture and modular design principles. This design reduces complexity by decomposing the system into manageable units, which improves development efficiency and long-term sustainability.

In conclusion, the architectural diagram not only visually represents the system structure but also reflects deeper design decisions regarding modularity, robustness, security, scalability, and maintainability. The separation of presentation, processing, and storage layers ensures that the developed QR code generation system in Python can operate efficiently under varying load conditions while maintaining compliance with encoding standards and security best practices.

The structural reliability of QR matrices implemented in this system is further supported by experimental findings reported in optical encryption research. Barrera et al. (2014) demonstrated that QR codes can function as robust information containers within joint-transform-correlator encryption architectures, allowing complete recovery of original information even after multiplexed optical scrambling and speckle-noise distortion. These findings confirm that the inherent redundancy and block-based organization of QR codes provide a resilient structural framework capable of

preserving data integrity under transformation. While the present implementation operates in a purely digital environment, this experimental evidence reinforces the architectural decision to maintain strict adherence to standardized encoding and error correction procedures.

By grounding the implementation strategy in experimentally validated QR robustness research, the developed system aligns not only with software engineering standards but also with interdisciplinary evidence demonstrating the resilience of QR-based information containers under transformation and noise conditions.

The diagram illustrates the potential integration pathway of the proposed QR code generation system within the national digital governance framework of the Kyrgyz Republic. The model reflects the interaction between educational institutions, state digital infrastructure, and the mobile application "Tunduk," enabling secure verification and digital document accessibility for citizens and administrative users

The presented conceptual model illustrates the potential integration pathway of the developed QR code generation system into the digital governance ecosystem of the Kyrgyz Republic. The diagram reflects a multi-level interaction framework, beginning with the software-based QR code generation module and extending through institutional and governmental digital infrastructures to end users, including citizens, students, and administrative personnel [8].

At the first level, the QR Code Generation System functions as a secure encoding and validation engine capable of producing standardized QR structures compliant with ISO/IEC 18004 requirements. At the second level, educational institutions (universities, schools, research centers) act as organizational nodes where generated QR codes may be embedded into academic documents, attendance systems, identification mechanisms, or internal administrative workflows.

The third level represents the State Digital Infrastructure, which encompasses national electronic interaction systems, digital registries, and interoperability platforms supporting data exchange between public institutions. Within this infrastructure, the mobile application "Tunduk" operates as a citizen-facing digital gateway that enables access to electronic documents and services in legally recognized digital format. According to the regulatory framework of the Kyrgyz Republic, digital documents accessed through "Tunduk" possess equivalent legal force to paper-based documents, reinforcing the practical significance of QR-based verification mechanisms.

Finally, at the user level, citizens, students, and administrators interact with digitally generated QR codes through mobile devices, facilitating secure verification, authentication, and information retrieval. The integration model demonstrates how a modular QR generation system can function as a technical component within broader digital transformation initiatives, supporting transparency, traceability, and interoperability in electronic governance environments [10].

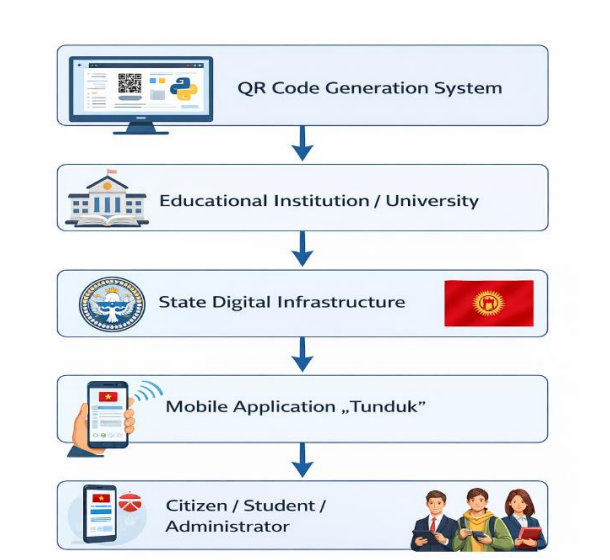


Figure 2. Conceptual Model of QR Code-Based Digital Document Flow within the Kyrgyz Republic's E-Government Infrastructure (including educational institutions and the "Tunduk" mobile platform)

The diagram therefore highlights not only the technical architecture of the system but also its contextual applicability within Kyrgyzstan's evolving e-government ecosystem. By aligning software implementation with national digital infrastructure principles, the proposed system contributes to scalable, secure, and legally compliant digital service delivery mechanisms.

In conclusion, the presented integration model demonstrates not only the technical feasibility of the developed QR code generation system but also its potential applicability within the digital governance infrastructure of the Kyrgyz Republic. The conceptual framework illustrates how the system can function as an intermediary component connecting educational institutions, state digital platforms, and the mobile application "Tunduk", thereby enabling secure verification and exchange of digital documents. Consequently, the proposed architecture reflects both algorithmic robustness and practical relevance, aligning with national digital transformation policies and supporting the advancement of electronic governance mechanisms.

References

1. Tribak, H., & Zaz, Y. (2017). QR Code Recognition based on Principal Components Analysis Method. International Journal of Advanced Computer Science and Applications (IJACSA), 8(4), 242–248. DOI: https://doi.org/10.14569/IJACSA.2017.080433. Available at: https://thesai.org/Publications/ViewPaper?Code=IJACSA&Issue=4&SerialNo=33&Volume=8. PDF: https://thesai.org/Downloads/Volume8No4/Paper_33-QR_%20Code_Recognition_based_on_Principal_Components.pdf
2. Kusyanti, A., & Arifin, A. (2017). QRishing: A User Perspective. International Journal of Advanced Computer Science and Applications (IJACSA), 8(10), 302–309. DOI: https://doi.org/10.14569/IJACSA.2017.081039. Available at: https://thesai.org/Publications/ViewPaper?Code=IJACSA&Issue=10&SerialNo=39&Volume=8. PDF: https://thesai.org/Downloads/Volume8No10/Paper_39-QRishing_a_User_Perspective.pdf
3. ISO/IEC. (2015). ISO/IEC 18004:2015 Information technology — Automatic identification and data capture techniques — QR Code bar code symbology specification. International Organization for Standardization. Available at (official ISO page): https://www.iso.org/standard/62021.html
4. Reed, I. S., & Solomon, G. (1960). Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics, 8(2), 300–304. https://doi.org/10.1137/0108018
5. Ohbuchi, E., Hanaizumi, H., & Hock, L. A. (2004). Barcode readers using the camera device in mobile phones. In Proceedings of the 2004 International Conference on Cyberworlds (pp. 260–265). IEEE. https://doi.org/10.1109/CW.2004.67
6. Lin, Y., & Fuh, C. S. (2013). 2D barcode image decoding based on adaptive thresholding and perspective correction. IEEE Transactions on Image Processing, 22(11), 4351–4364. https://doi.org/10.1109/TIP.2013.2278010
7. ZXing Project. (2024). Zebra Crossing: Multi-format 1D/2D barcode image processing library. Retrieved from https://github.com/zxing/zxing
8. Li, J., Bian, W., Diao, Y., Zou, T., Yang, X., & Kang, B. (2025). A Multi-User Quiz System with QR Code Identification and Configurable Timing. Applied System Innovation, 8(6), 158. https://doi.org/10.3390/asi8060158 https://www.mdpi.com/2571-5577/8/6/158
9. Pressman, R. S., & Maxim, B. R. (2019). Software Engineering: A Practitioner's Approach (9th ed.). McGraw-Hill Education. https://www.mheducation.com/highered/product/software-engineering-practitioner-s-approach-pressman/M9781259872976.html
10. Larman, C. (2004). Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd ed.). Prentice Hall. https://www.informit.com/store/applying-uml-and-patterns-9780131489066
11. Sommerville, I. (2011). Requirements engineering: A good practice guide. Wiley. https://onlinelibrary.wiley.com/doi/book/10.1002/9780470725764

Рецензент: к. социологич. н., доцент Шайылдаева А.К.