International Journal of Business and Technology Management
e-ISSN: 2682-7646 | Vol. 7, No. 2, 391-407, 2025
https://asianscholarsnetwork.com/asnet-journals

A Collaborative Software Development Model for Citizen Developers to Address Growing Complexity of Workforce Challenges

Imran Harith Azmy[1*], Azri Azmi[1], Nazri Kama[1], Hazlifah Mohd Rusli[1]

[1] Faculty of Artificial Intelligence, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia

*Corresponding Author: imranharithazmy@graduate.utm.my

Received: 4 January 2025 | Accepted: 25 March 2025 | Published: 1 April 2025

DOI: https://doi.org/10.55057/ijbtm.2025.7.2.34

Abstract: *The growing complexity of software development, combined with the rising demand for software products, has led to a shortage of skilled programmers, causing difficulties in meeting project timelines. Citizen Development has emerged as a potential solution by involving non-developers in the development process. This study explores Software Engineers' views on Citizen Development through a detailed survey, uncovering a generally positive outlook on its potential. Based on these insights, a new software development model was designed and evaluated using the System Usability Scale (SUS). The results suggest that this model could facilitate the transformation of the software development landscape by making application development more accessible through empowering Citizen Developers and helping to alleviate the software engineering talent shortage.*

Keywords: Project Management, Citizen Development, Citizen Developers, Non-Programmers, Rapid Application Development.

---

1. Introduction

Traditional software development has historically depended on highly skilled experts to manage increasingly complex systems (Azmy et al., 2022). Professional developers face substantial demands driven by the industry's competitive landscape, which amplifies the urgency to produce results quickly results (Alsaadi et al., 2021; Silva et al., 2020; Waszkowski, 2019; Woo, 2020). Moreover, a recent forecast from Forrester Research, a prominent global market research firm, projects that the United States will experience a shortage of more than 500,000 software developers in the near future (Project Management Institute, 2021; Singh, 2021).

Adopting Citizen Development effectively is essential because it has the capacity to alleviate the workforce challenges surrounding software development, which is widely acknowledged. Because citizen development differs from conventional software development in terms of development procedures, associated risks, and obstacles, it is anticipated that current software development models may be less appropriate for the activity. A software development model designed especially for Citizen Developers must be created in order to guarantee the effective implementation of Citizen Development and to optimise its advantages. This strategy will give organisations a methodical and efficient way to empower their workers, leverage the knowledge of non-programmers, and improve software development skills overall. By

harnessing the domain knowledge and creativity of non-programmers, organizations can enhance their software development capacity while easing the burden on professional developers. Experts predict that Citizen Developers may eventually outnumber professional software developers (Martinez & Pfister, 2023; Wong et al., 2019).

2. Literature Review

Individuals with extensive knowledge or skill of programming and software development, or those who have obtained formal education in a software-related discipline at the degree or professional level, are classified as 'programmers' (Namoun et al., 2019). Conversely, individuals who have not studied software development or related fields, lack prior programming experience, or are entirely new to programming are categorized as 'non-programmers' (Namoun et al., 2019). The increasing use of automated tools for application creation, deployment, and maintenance has significantly reduced the technical expertise required for software development. As a result, even individuals with minimal experience can now develop platform-specific applications.

In recent years, the term 'Citizen Developers' has gained prominence in the IT industry as a more precise label than 'non-programmers' to describe individuals with little to no background in programming or software engineering (Oltrogge et al., 2018; Tisi et al., 2019). Citizen developers are typically business-oriented power users who collaborate with IT teams and professional developers to create software, integrations, automation solutions, business intelligence/analytics tools, or artificial intelligence (AI) models (Salgueiro, 2021).

2.1 Increasing Relevance of Citizen Developers

A survey conducted by Thacker et al. (2021) among business students revealed that while many initially showed reluctance, they were ultimately enthusiastic about the opportunity to work as technology developers. This hesitation stemmed from their lack of technical knowledge and hands-on experience. The study highlights a significant opportunity: non-programmers, including business students, often have a strong desire to create digital solutions but face barriers due to their limited technical expertise (Thacker et al., 2021). Individuals in business roles or with business education could become effective 'Citizen Developers' — especially if they possess a solid understanding of business processes — enabling them to design digital solutions that efficiently address business challenges.

Hoogsteen and Borgman (2022) explored factors influencing the adoption of citizen development within organizations, particularly decisions regarding end-user involvement in software development (Hoogsteen and Borgman, 2022). Their findings suggest that the drive for digital transformation — fueled by the need to modernize internal systems to keep up with fast-changing market demands — has intensified the demand for software and developers (Alt et al., 2020). With IT departments overwhelmed by growing backlogs, organizations are encouraged to embrace citizen development. However, the study identifies key dependencies, including types of end users, organizations, and technologies, and recommends selecting appropriate use cases to avoid integration issues while still requiring minimal technical skills (Hoogsteen and Borgman, 2022).

Moreover, citizen developers can eliminate the need for intermediaries between business and IT teams, accelerating prototype development and enhancing productivity by reducing miscommunication. Oltrogge et al. (2018) suggest that empowering citizen developers can streamline software development, automate processes, and incorporate business insights

directly into solutions. This approach ensures ideas are quickly translated into functional applications (Oltrogge et al., 2018). However, it's crucial to equip the workforce with intuitive, powerful technological tools — especially as today's workers, having grown up in a digital-first world, are highly collaborative, tech-savvy, and expect user-friendly, robust technology (Tisi et al., 2019). Additionally, many IT departments and businesses face challenges in providing non-technical employees with the right tools to harness their potential and address the ongoing developer shortage. Viewed from this perspective, citizen developers represent an opportunity for organizations to adapt to evolving workforce and technology trends, unlocking untapped talent and fostering innovation.

Several studies and forecasts indicate that Citizen Developers will be responsible for creating a significant share of future applications. A study, through a survey discovered that over 70 percent of respondents acknowledged employees outside their IT departments were developing technology solutions using Citizen Development tools (Lebens & Finnegan, 2021). Vincent et al. (2019) projects that 65 percent of applications will be built by Citizen Developers, supporting various use cases such as reporting, analysis, event processing, user interfaces, data services, and business logic. (Vincent et al., 2019). Additionally, more than 90 percent of large organizations plan to adopt or are exploring Citizen Development platforms in the near future (Martinez & Pfister, 2023). Another global survey conducted also revealed that over 75 percent of organizations have already adopted no-code/low-code platforms (Project Management Institute, 2021).

2.2 Citizen Development Method: Low-Code Development

Arora et al. discussed Sagitec Software Studio (S3), a low-code development platform by Sagitec Solutions LLC that allows users to build applications through a 'drag and drop' interface with minimal coding, which is actively used to create enterprise-level applications for industries like healthcare, insurance, and pensions (Arora et al., 2020). Similarly, Aurea Business Process Modelling (BPM) is a low-code platform designed to automate business processes in the manufacturing sector (Waszkowski, 2019). SetXRM, another low-code platform, offers practical and flexible solutions for short-term needs (Sahinaslan et al., 2021). Additionally, three widely recognized low-code platforms — Microsoft PowerApps, Mendix, and OutSystems — were analyzed for their capabilities (Gurcan & Taentzer, 2021). Microsoft PowerApps is a cloud-based platform that focuses on data handling, logic, UI design, and visualization with minimal programming. Mendix supports mobile and web application development and offers a collaborative integrated development environment, while OutSystems, like PowerApps, is a cloud-based platform that facilitates both web and mobile app development (Gurcan & Taentzer, 2021).

However, there are many low-code platforms are limited in customization due to their "What You See Is What You Get" (WYSIWYG) design approach, which restricts flexibility (Gurcan & Taentzer, 2021). Furthermore, there is an argument that low-code platforms may struggle to adapt to evolving technical requirements (Woo, 2020). Meanwhile, there are concerns about potential security vulnerabilities within these platforms (Oltrogge et al., 2018).

2.3 Citizen Development Method: No-Code Development

In contrast to low-code development, AppSheet has been recognized as a prominent and promising no-code development platform that requires no programming during the development process (Di Ruscio et al., 2022). Research and testing conducted on micro, small, and medium-sized businesses using AppSheet indicate that the technology has significant positive impacts, particularly by substantially reducing costs and development time

(Nurharjadmo et al., 2022). However, many publications use the terms 'no-code' and 'low-code' interchangeably or combine them as ‘low/no-code’ (Bhattacharyya & Kumar, 2021; Yan, 2019). Di Ruscio et al. (2022) argue that ‘no-code development platform’ refers specifically to platforms that require no programming, visual languages, configurations, or graphical user interfaces. Despite this, market analysis firms often resist classifying it as a separate market segment, resulting in ‘low-code development platform’ being the more commonly — and sometimes exclusively — used term (Di Ruscio et al., 2022). Therefore, this study only considers literature that explicitly defines the technology as ‘no-code development.’

2.4 Citizen Development Method: Spreadsheet-Driven Development

'Quilt' has been described as a responsive read-write system that enables the creation of basic web applications by using spreadsheets as servers and HTML for defining the client-side user interface (Benson et al., 2014). Meanwhile, the 'Gneiss System' was developed to combine a web interface builder with a spreadsheet editor, allowing end-users to link web GUI elements with spreadsheet cell attributes to manage the application’s service data (Shih et al., 2017). However, a key limitation of spreadsheet-driven development is the absence of a clear, effective method to connect spreadsheets to the internet, which restricts the development of robust data management systems (Yang et al., 2019).

2.5 Citizen Development Method: Automatic Code Generator

Automatic code generators play a crucial role in accelerating application development by simplifying the creation of common features, such as CRUD-based functionality. A technique for automated code generation was introduced, leveraging Model Driven Architecture and the ‘Develop Once, Run Everywhere’ principle to streamline and speed up the development process (Benouda et al., 2018). Moreover, a model-driven framework designed for cross-platform web and mobile applications was proposed, generating scaffolding code for CRUD operations. This framework includes a Domain Specific Modeling Language (DSML) built on the Unified Modeling Language (UML), supporting various data types and stereotypes (Inayatullah et al., 2019).

On the other hand, an open-source transformation engine was developed to autonomously generate scaffolding CRUD code from high-level class diagrams for cross-platform applications. A code automation tool, Re-CRUD, was created to produce a complete and integrated electronic records management system tailored for web application development (Anuar et al., 2022). When tested on a small-scale web application, the Electronic Database Management System (EDMS), Re-CRUD demonstrated greater efficiency and cleaner code generation compared to traditional frameworks like CakePHP, Laravel, Symphony, and FuelPHP (Anuar et al., 2022).

3. Methodology

To achieve the purpose of this study, a structured research methodology has been designed and executed. This methodology follows a clear, systematic sequence of steps to ensure the results are reliable and credible. Figure 1 illustrates the complete methodology.

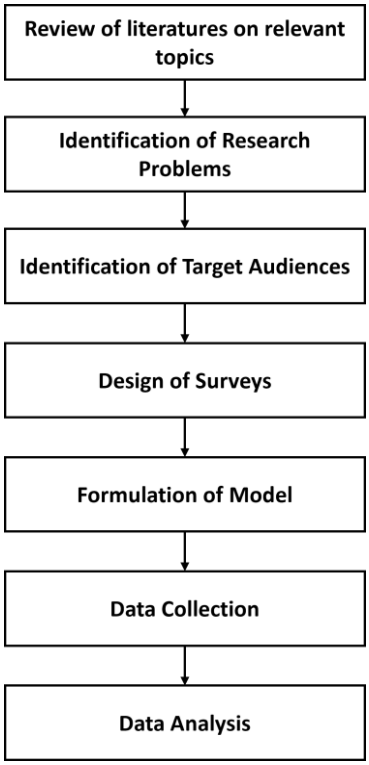

Figure 1: Research Methodology

The study begins with a comprehensive review of relevant literature, focusing on key issues. This review critically analyzes essential topics, including the definition of Citizen Developer, exploring its conceptual framework and clarifying its characteristics as presented in academic discussions. It also examines the emergence of Citizen Developers, offering an in-depth analysis of past and current trends that shed light on their evolution in software development. Additionally, the study investigates the significant role Citizen Developers play in the software development landscape, emphasizing their responsibilities, contributions, and overall impact. Finally, it evaluates various methods and strategies associated with Citizen Development, providing a clear explanation of their core principles and processes.

Next, the study identifies and defines a key research problem. This issue highlights the ongoing shortage of Software Engineers in the industry and proposes that integrating Citizen Development practices could help alleviate this gap. The concept suggests that Citizen Developers could collaborate with and support traditional development teams. As a result, the study aims to evaluate Software Engineers' readiness and willingness to embrace this approach.

3.1 Methodology: Sentiment Survey

The sentiment survey specifically targets individuals identifying as Software Engineers, whether through professional experience or formal academic qualifications. This criterion ensures that the study gathers insights from participants with a strong understanding of software development processes.

To collect relevant data and insights for the sentiment survey, a structured questionnaire has been developed, consisting of two main sections. The first section focuses on Demographic Information, featuring six questions designed to capture key participant characteristics. The second section includes eleven Specific Questions aimed at gathering detailed opinions and

perceptions regarding the integration of Citizen Developers into development teams. Table 1 provides a summary of the Specific Questions included in this questionnaire.

Table 1: List of Specific Questions for Sentiment Survey

Question ID	Specific Question	Insights provided
SQ1	At this point, how familiar are you with the concept of 'Citizen Development'?	Familiarity with Citizen Development
SQ2	Have you ever worked with or collaborated with non-programmers (Citizen Developers) during software development process?	Experience with Non-Programmers in Software Development
SQ3	In your current organization, are there any active Citizen Development initiative?	Presence of Active Citizen Development Initiatives
SQ4	In your opinion, to what extent Citizen Developers should be involved in software development?	Extent of Involvement of Citizen Developers
SQ5	In your opinion, how helpful do you think it is for Citizen Developers to assist software developers at any stage of the software development process?	Perceived Helpfulness of Citizen Developers
SQ6	In your opinion, what types of applications do you think are suitable for Citizen Developers to be involved in?	Types of Applications Suitable for Citizen Developers
SQ7	Do you believe that involving Citizen Developers in the software development team can free up software developers' time to focus on more complex projects and tasks?	Impact on Software Developers' Time
SQ8	In your opinion, how would you rate the effectiveness of having Citizen Developers assist software developers in the software development process?	Effectiveness of Citizen Developers' Assistance
SQ9	In your opinion, what would be the ideal combination between software developers and citizen developers in a software development team?	Ideal Combination in a Software Development Team
SQ10	In your opinion, should Citizen Development have a model, framework or guideline to effectively develop or assist in developing software?	Need for a Model or Framework
SQ11	Overall, do you think that the concept of Citizen Development would promote manpower optimisation in an organisation or company?	Promotion of Manpower Optimization

Additionally, the data collection process employs a carefully designed random sampling approach to ensure the inclusion of individuals from the defined target population. This method strengthens the reliability and relevance of the collected data, enhancing the generalizability of the findings.

Following data collection, a thorough and systematic analysis is conducted. This stage involves interpreting and synthesizing the data to develop a comprehensive understanding of Software Engineers' attitudes, perspectives, and readiness regarding the Citizen Development concept. The results of this analysis will be detailed in the following sections, offering valuable insights and implications that contribute to advancing knowledge in this field.

3.2 Methodology: System Usability Scale (SUS)

SUS is widely used for evaluating perceived usability, with over 5,000 citations reported by 2018 (Lewis, 2018). It is favored for its simplicity, speed, and ease of administration, making it accessible even when participants face challenges during testing (Drew et al., 2018). The questionnaire alternates between positive and negative statements to reduce bias from fatigue or inattention, producing a single usability score (Drew et al., 2018). The following Table 2 outlines the specific questions for SUS in this study. Each respondent has to rate between 1 and 5 based on their agreement with the survey item statement.

Table 2: List of Specific Questions for SUS

Question ID	Survey Item Statement	Aspect
SUS1	The platform's features support me in meeting software development requirements.	Effectiveness
SUS2	I doubt I can create a high-impact application using this model.	
SUS3	I can effectively communicate my ideas to other developers while using this platform.	
SUS4	The platform's features feel limited; I need additional functionalities.	
SUS5	I found this platform easy to use.	Efficiency
SUS6	I think the steps involved are complicated to follow.	
SUS7	The platform doesn't require extensive learning for me to complete tasks.	
SUS8	The platform's user interface and experience feel unintuitive.	
SUS9	I can complete tasks without needing extra knowledge of software development management.	
SUS10	I feel uncomfortable using this platform.	

To gain a clearer understanding of the proposed solution's evaluation, the standard SUS grading scale is referred as a guide, that is depicted in the following Figure 2 (Adobe, 2020).

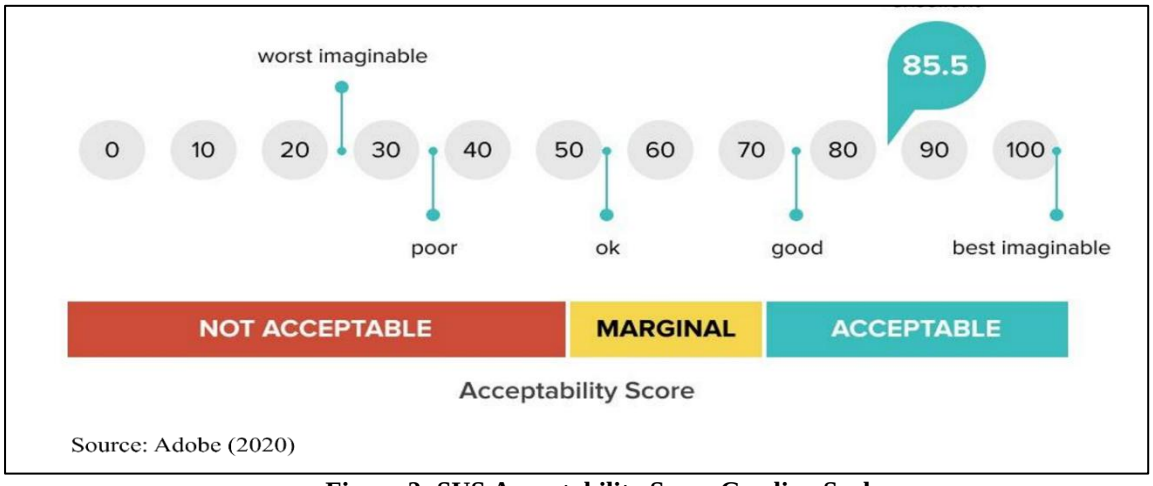

Figure 2: SUS Acceptability Score Grading Scale

4. Sentiment Survey: Findings and Analysis

A total of twenty individuals who identify as software developers participated in the survey, which was distributed randomly. Demographic information — including gender, age, highest education level, years of experience in software development, organization size, and industry type — was collected to better understand their backgrounds.

The demographic breakdown of participants showed that 75 percent were male and 25 percent were female. In terms of age, participants ranged from 18 to 34 years old, with the majority falling within the 25 to 34 age group. Regarding education, 75 percent held a bachelor's degree as their highest qualification. The participants' software development experience varied between 1 and 10 years, averaging around 3 years. They represented organizations of different sizes, from small businesses with 1-50 employees to large companies with over 501 employees, with a notable presence of participants from smaller organizations. Additionally, 70 percent of the participants worked in the Technology or IT sector, making it the most represented industry in the study.

The survey comprised eleven targeted questions focusing on Citizen Development, designed to gather insights based on respondents' software development experience. The goal of this research was to better understand the readiness and acceptance of software engineers toward Citizen Development — an emerging approach that aims to alleviate the ongoing software engineer shortage by involving non-developers in the development process.

This section presents the perspectives and insights shared by the professionals who participated in the survey. Their feedback offers valuable understanding of how Citizen Development is perceived within the software development community, setting the stage for a deeper discussion on its potential benefits and challenges. Figure 3 provides a summary of the questionnaire results, with detailed analysis explained in the following sections.

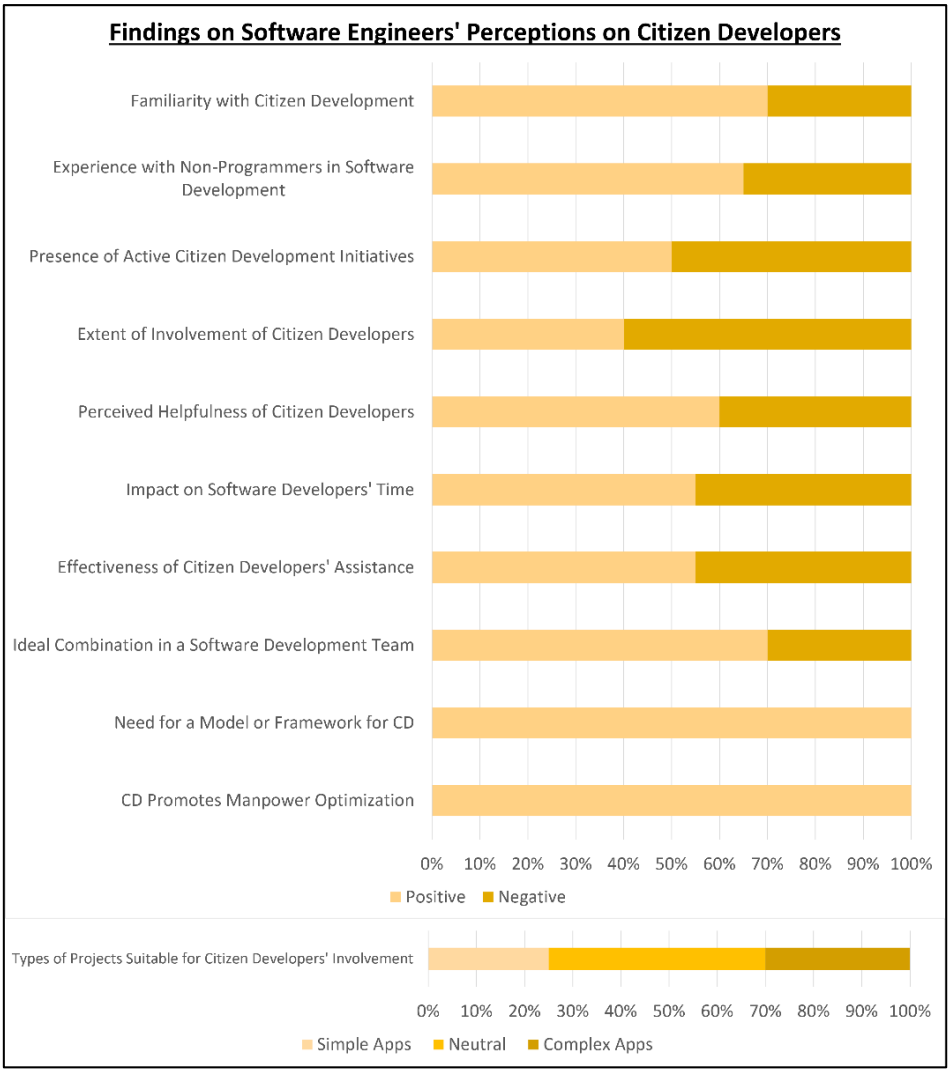

Figure 3: Sentiment Survey Findings Summary

The survey findings reveal a notable level of familiarity with the concept of Citizen Development among software developers, with 70 percent of respondents acknowledging awareness of the practice. This indicates that Citizen Development is not only a recognized term but potentially a growing trend within the software development landscape. Furthermore, 65 percent of participants reported prior collaboration with non-programmers — Citizen Developers — suggesting an increasing integration of such individuals into software development teams. This shift points towards a broader acceptance of diverse team compositions, where Citizen Developers contribute alongside professional developers.

Half of the respondents indicated that their organizations already have active Citizen Development initiatives, reflecting a moderate but significant level of adoption. This suggests that many companies are actively exploring the potential of Citizen Development. Additionally, 40 percent of participants expressed a belief that Citizen Developers should be maximally involved in development processes, demonstrating confidence in their ability to support and enhance software projects. Reinforcing this perspective, 60 percent of respondents affirmed the helpfulness of Citizen Developers, recognizing their potential to boost productivity and efficiency across various stages of development.

Opinions regarding the types of projects suitable for Citizen Development varied, with 25 percent endorsing simple applications and 30 percent believing Citizen Developers could

contribute to complex projects. Despite these differences, the data suggests a growing belief that Citizen Developers can handle projects of varying complexity. Moreover, 55 percent of respondents agreed that integrating Citizen Developers into development teams can alleviate the workload of professional developers, allowing them to concentrate on more complex tasks. This highlights the potential of Citizen Development to optimize resource allocation and improve time management within development teams.

The survey also indicates that 55 percent of participants view Citizen Developers' involvement as effective, particularly when working alongside professional developers — suggesting that collaborative teams yield positive outcomes. Notably, 70 percent of respondents supported a team composition with a majority of professional developers complemented by a minority of Citizen Developers, reflecting a preference for hybrid teams that leverage the strengths of both groups.

A particularly striking result emerged regarding the need for structure: all participants unanimously agreed that Citizen Development requires a well-defined model, framework, or guideline to ensure its successful implementation. This consensus emphasizes the importance of clear processes to support and guide Citizen Developers within development environments. Additionally, 100 percent of respondents agreed that Citizen Development promotes workforce optimization, reinforcing the idea that this approach has the potential to enhance organizational efficiency and resource management.

In conclusion, the survey findings present a compelling case for the growing recognition and adoption of Citizen Development within software development teams. While perspectives on the extent of Citizen Developers' involvement and the types of projects they can handle vary, the overall sentiment leans towards acceptance and optimism. The unanimous agreement on the need for a structured framework further underscores the importance of strategic implementation. Ultimately, Citizen Development emerges not only as a viable solution to address the software developer shortage but also as a promising approach to optimize workforce capabilities and improve software development outcomes.

5. Proposed Solution

Collaboration between professional developers and end users or domain experts plays a crucial role in ensuring effective software requirement fulfillment. By empowering end users or domain experts as Citizen Developers, project teams can leverage their domain knowledge to enhance the development process. Their direct and active participation in software creation necessitates a supportive and well-structured collaborative environment. Such an environment can be established through well-defined development processes, typically represented by a software development model (SDM) or software development lifecycle (SDLC).

In response to these challenges, a software development model has been proposed to facilitate a structured workflow for the collaborative development of software products. This model is designed not only to streamline the development process but also to address the complexities of software requirements by bridging the gap between technical expertise and domain knowledge.

5.1 The Collaborative Model

The proposed software development model is built upon the core stages defined in the SWEBOK Guide V4.0 (IEEE Computer Society, 2024) and aims to address gaps identified in

Microsoft's Agile v2 Model for Citizen Development (Microsoft, 2022) and PMI's Hyperagile Model (Project Management Institute, 2021). Traditional software development models typically follow six key stages: initiation, requirements, design, coding, testing, deployment, and maintenance (Han et al., 2020). These stages serve as the structural foundation of the proposed model.

Furthermore, given that Microsoft Agile v2 and PMI Hyperagile prioritize prototyping methodologies, the proposed model integrates the Prototype Software Development Model (SDM). This approach focuses on rapidly capturing initial user requirements through collaboration between developers and users, facilitating the quick development of a working prototype. The prototype is then refined through iterative cycles of user feedback and evaluation until it aligns with user expectations and needs (Yu, 2018).

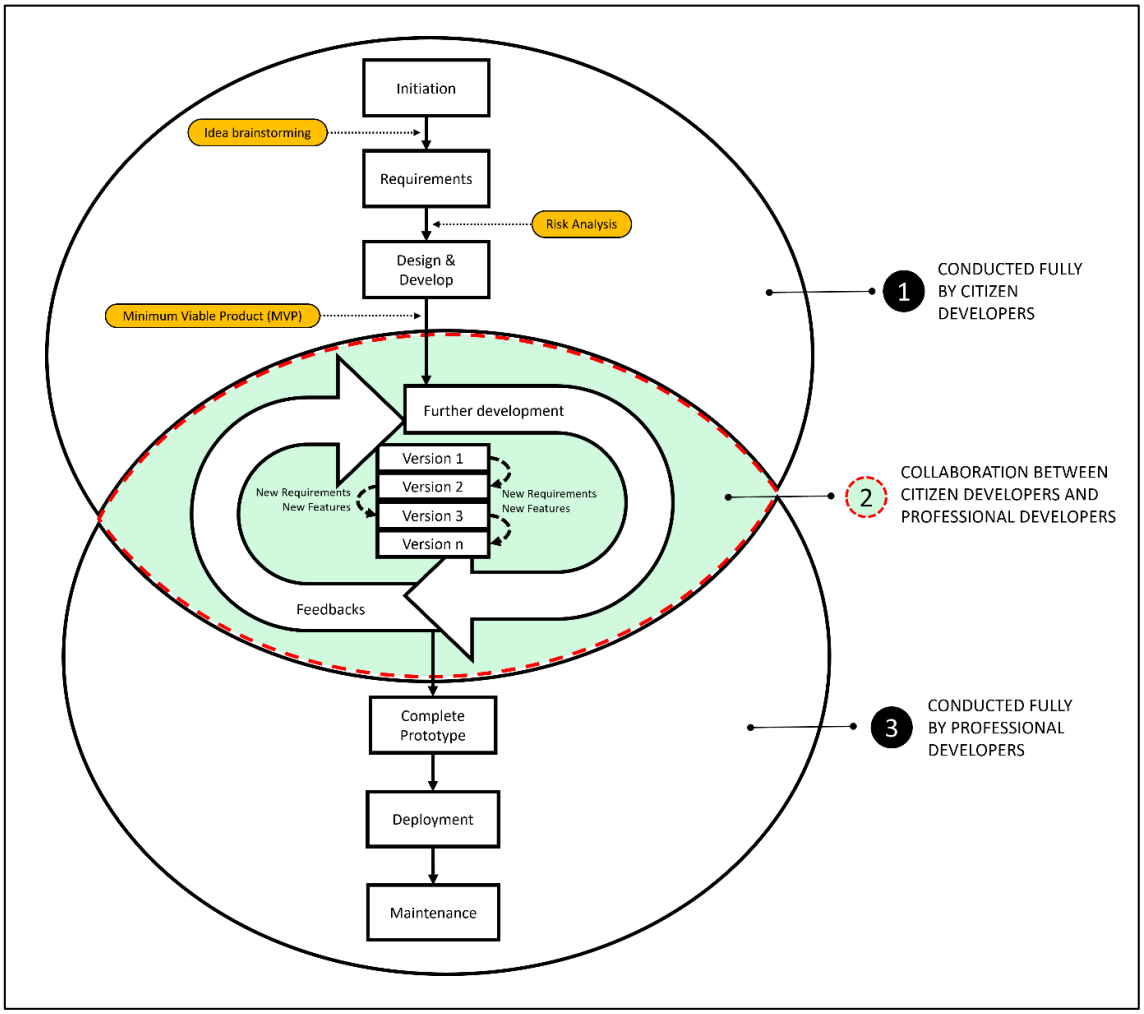

Figure 4: Proposed Model

The proposed model consists of three key components: citizen development, collaboration, and conventional development. In the citizen development component, the process starts with the requirements phase, where citizen developers initiate the project by outlining initial ideas and collecting high-level requirements for further analysis. These requirements are then categorized and prioritized based on their complexity, helping to define roles and responsibilities. A risk analysis is also performed to evaluate whether the project is feasible and should proceed. The design phase within this component centers on building a minimum viable product (MVP), focusing on creating a functional foundation for the software.

After the design phase, the development phase involves professional software developers — referred to as "conventional developers" — who step in to handle technical challenges that exceed the citizen developers' capabilities. These developers ensure that all requirements are fully addressed and work collaboratively with citizen developers, offering technical expertise and guidance throughout the process. This iterative, cooperative approach ensures a seamless progression from design to development, ultimately resulting in a well-rounded, functional application.

6. Validation on Proposed Solution

The proposed model was validated using the System Usability Scale (SUS), a widely recognized method for assessing the usability and effectiveness of software systems. To ensure practical evaluation, a web-based Collaborative Platform was specifically developed to serve as a demonstration of the model's functionality. This platform was designed to simulate a real-world scenario where citizen developers and conventional developers collaborate throughout the software development process. The platform showcased key features of the model, such as requirements gathering, risk analysis, design, and development phases, enabling users to experience the workflow first-hand. By interacting with the platform, respondents could engage with the collaborative environment and explore how the model supports citizen developer involvement alongside professional developers.

A total of 10 citizen developers participated as respondents, testing the platform and providing feedback through the SUS questionnaire. This approach measured their overall experience, focusing on ease of use, efficiency, and perceived effectiveness of the collaborative model. The SUS score, ranging from 0 to 100, provided a quantitative measure of the model's usability.

6.1 System Usability Scale (SUS)

Ten respondents, carefully chosen to align with the Citizen Developer criteria, participated in the evaluation. These individuals came from diverse industries, including Health, Safety, and Environment (HSE), Energy, Finance and Banking, and Manufacturing. Their varied professional backgrounds provided a broader perspective on the model's practicality and usability across different sectors. The respondents had professional experience ranging from 2 to 12 years, representing a balanced mix of early-career professionals and seasoned experts, as summarised in Table 3. Their combined feedback, gathered through the SUS questionnaire, provided essential quantitative data to assess and validate the model's design and effectiveness.

Table 3: SUS Respondents

Respondent	Working Experience	Main Industry/ Domain	Citizen Development Platform
CD1	9 years	HSE	• Microsoft Power Platform
CD2	3 years	Security and Surveillance	• Microsoft Power Platform • Google AppSheet
CD3	12 years	Energy	• Microsoft Power Platform • Google AppSheet
CD4	6 years	Oil and Gas	• Microsoft Power Platform
CD5	3 years	Energy	• Microsoft Power Platform
CD6	7 years	Education Technology	• Microsoft Power Platform • Google AppSheet
CD7	5 years	Energy	• Outsystem • Microsoft Power Platform
CD8	4 years	Manufacturing	• Microsoft Power Platform • Google AppSheet
CD9	2 years	Finance and Banking	• Microsoft Power Platform
CD10	3 years	Oil and Gas	• Microsoft Power Platform

6.2 Results and Discussion

The System Usability Scale (SUS) results offer a thorough evaluation of the platform's usability, with an overall score of 70.25. This score signifies an acceptable level of performance while also revealing areas that need improvement — particularly to align with the needs of Citizen Developers, who value simplicity and ease of use.

Table 4: SUS Results

ID/Aspect	Effectiveness				Efficiency					
	SUS 1	SUS 2	SUS 3	SUS 4	SUS 5	SUS 6	SUS 7	SUS 8	SUS 9	SUS 10
CD 1	4	2	4	2	2	2	4	3	4	2
CD 2	5	2	4	2	5	1	4	2	4	1
CD 3	5	1	4	3	5	1	4	1	5	2
CD 4	5	1	4	3	5	1	4	1	5	2
CD 5	4	3	4	2	4	4	3	4	3	3
CD 6	4	3	4	4	4	3	4	3	4	3
CD 7	3	2	4	2	4	2	3	2	3	2
CD 8	3	4	4	3	3	3	4	4	3	3
CD 9	4	3	4	4	4	1	4	1	3	1
CD 10	4	1	5	4	4	5	4	5	2	3
SUS Score (by Question)	80.0	70.0	77.5	57.5	75.0	67.5	72.5	60.0	67.5	75.0
SUS Score (by Aspect)	71.25				69.58					
SUS Score (Overall)	70.25									

The platform's effectiveness is evident with a SUS score of 71.25, showcasing its ability to help users achieve their software development goals. It performs particularly well in supporting requirement fulfillment (SUS1: 80) and enabling collaboration (SUS3: 77.5), both essential for the proposed Collaborative Model. However, a major weakness lies in the platform's feature set, reflected in SUS4's lower score of 57.5, suggesting that users find the existing features insufficient for handling more complex tasks. This limitation hampers the platform's overall effectiveness, highlighting a need for expanded capabilities to better support advanced development processes.

In terms of efficiency, the platform scored a moderate 69.58, indicating that while it remains functional, improvements are necessary — especially in design and workflow accessibility. Strengths include ease of use (SUS5 and SUS10: 75) and a low learning curve (SUS7: 72.5), making it intuitive and accessible for Citizen Developers. However, the user interface and overall user experience scored lower (SUS8: 60), pointing to dissatisfaction with the platform's design. Additionally, the need for more software development management knowledge emerged as a concern (SUS9: 67.5), suggesting that Citizen Developers may feel underprepared when collaborating with professional developers. Despite this, practical experience and continued exposure to the model's collaborative approach could gradually address this confidence gap. To ensure long-term success, improvements in feature depth, user experience, and tailored support for Citizen Developers will be essential.

Overall, the platform demonstrates a solid foundation in enabling Citizen Developers to participate in software development through its strengths in ease of use, low learning curve, and collaborative features. However, enhancing the platform's feature set, refining its interface, and offering tailored support to boost Citizen Developers' confidence in managing

development tasks will be crucial for maximizing its effectiveness and ensuring long-term success in collaborative software development environments.

7. Conclusion

In conclusion, the survey analysis reveals optimism among Software Engineers about integrating Citizen Developers into the software development process, highlighting the potential for a more collaborative approach. To address identified gaps, a new software development model was proposed, blending the strengths of Citizen Developers and conventional developers within a structured, inclusive framework. While the study has limitations that may affect the precision of its findings, the insights gained are valuable for understanding the role of Citizen Development. The model, evaluated through the System Usability Scale (SUS) with practicing Citizen Developers, demonstrated promise in making software development more accessible and addressing the software engineering talent shortage. Future efforts will focus on refining use cases, enhancing implementation processes, and conducting thorough validation to ensure the model's practicality and adaptability in real-world environments — contributing to more innovative, efficient, and sustainable software development practices.

Acknowledgements
This work was supported/funded by the Ministry of Higher Education of Malaysia under Fundamental Research Grant Scheme (FRGS/1/2020/ICT01/UTM/02/1) (Vote No. R.K130000.7856.5F415).

References

Adobe. (2020). *The System Usability Scale & How It's Used in UX*. Https://Xd.Adobe.Com/Ideas/Process/User-Testing/Sus-System-Usability-Scale-Ux/.

Alsaadi, H., Radain, D., Alzahrani, M., Alshammari, W., Alahmadi, D., & Fakieh, B. (2021). *Factors that affect the utilization of low code development platforms: survey study*.

Alt, R., Leimeister, J. M., Priemuth, T., Sachse, S., Urbach, N., & Wunderlich, N. (2020). Software-Defined Business: Implications for IT Management. *Business and Information Systems Engineering*, *62*(6), 609–621. https://doi.org/10.1007/s12599-020-00669-6

Anuar, A. W., Kama, N., Azmi, A., Rusli, H. M., & Yahya, Y. (2022). *Re-CRUD Code Automation Framework Evaluation using DESMET Feature Analysis*. *13*(5), 437–452.

Arora, R., Ghosh, N., & Mondal, T. (2020). Sagitec Software Studio (S3) - A Low Code Application Development Platform. *2020 International Conference on Industry 4.0 Technology (I4Tech)*, 13–17. https://doi.org/10.1109/I4Tech48345.2020.9102703

Benouda, H., Azizi, M., Moussaoui, M., & Esbai, R. (2018). Automatic code generation within MDA approach for cross-platform mobiles apps. *Proceedings of EDIS 2017 - 1st International Conference on Embedded and Distributed Systems*, *2017-Decem*, 1–5. https://doi.org/10.1109/EDIS.2017.8284045

Benson, E., Zhang, A. X., & Karger, D. R. (2014). Spreadsheet-driven web applications. *UIST 2014 - Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, 97–106. https://doi.org/10.1145/2642918.2647387

Bhattacharyya, S. S., & Kumar, S. (2021). Study of deployment of "low code no code" applications toward improving digitization of supply chain management. *Journal of*

Science and Technology Policy Management. https://doi.org/10.1108/JSTPM-06-2021-0084

Di Ruscio, D., Kolovos, D., de Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Low-code development and model-driven engineering: Two sides of the same coin? *Software and Systems Modeling*, *21*(2), 437–446. https://doi.org/10.1007/s10270-021-00970-2

Drew, M. R., Falcone, B., & Baccus, W. L. (2018). What Does the System Usability Scale (SUS) Measure? In A. Marcus & W. Wang (Eds.), *Design, User Experience, and Usability: Theory and Practice* (pp. 356–366). Springer International Publishing.

Gurcan, F., & Taentzer, G. (2021). Using Microsoft PowerApps, Mendix and OutSystems in Two Development Scenarios: An Experience Report. *Companion Proceedings - 24th International Conference on Model-Driven Engineering Languages and Systems, MODELS-C 2021*, 67–72. https://doi.org/10.1109/MODELS-C53483.2021.00017

Han, D., Li, X., Zhang, J., Wang, G., Deng, H., & Deng, L. (2020). *Research and Application of Software Development Model in Computer Software Development*. https://doi.org/10.1145/3561877

Hoogsteen, D., & Borgman, H. (2022). Empower the Workforce, Empower the Company? Citizen Development Adoption. *Proceedings of the 55th Hawaii International Conference on System Sciences*. https://hdl.handle.net/10125/79912

IEEE Computer Society. (2024). *Guide to the Software Engineering Body of Knowledge Version 4.0 (SWEBOK Guide V4.0)*. https://waseda.app.box.com/v/ieee-cs-swebok

Imran Harith Azmy, Azri Azmi, Nazri Kama, Hazlifah Mohd Rusli, & Asyraf Wahi Anuar. (2022). Towards a Semantic-Based Searching In a Web Application Framework For an Improved Development Productivity and Usability. *Journal of Information System and Technology Management*, *7*(28), 12–23. https://doi.org/10.35631/JISTM.728002

Inayatullah, M., Azam, F., & Anwar, M. W. (2019). Model-Based Scaffolding Code Generation for Cross-Platform Applications. *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2019*, 1006–1012. https://doi.org/10.1109/IEMCON.2019.8936289

Lebens, M., & Finnegan, R. (2021). *Rise of the Citizen Developer* (Vol. 5).

Lewis, J. R. (2018). The System Usability Scale: Past, Present, and Future. *International Journal of Human–Computer Interaction*, *34*(7), 577–590. https://doi.org/10.1080/10447318.2018.1455307

Martinez, E., & Pfister, L. (2023). Benefits and limitations of using low-code development to support digitalization in the construction industry. In *Automation in Construction* (Vol. 152). Elsevier B.V. https://doi.org/10.1016/j.autcon.2023.104909

Microsoft. (2022). *Differences between Power Apps and traditional app development approaches*. https://learn.microsoft.com/en-us/power-apps/guidance/planning/app-development-approaches

Namoun, A., Owrak, A., & Mehandjiev, N. (2019). Non-programmers composing software services: A confirmatory study of the mental models and design challenges. *Applied Sciences (Switzerland)*, *9*(24). https://doi.org/10.3390/app9245558

Nurharjadmo, W., Khadija, M. A., & Wahyuning, T. (2022). Modern No Code Software Development Android Inventory System for Micro, Small and Medium Enterprises. *Proceedings - 2022 IEEE International Conference on Cybernetics and Computational Intelligence, CyberneticsCom 2022*, 191–195. https://doi.org/10.1109/CyberneticsCom55287.2022.9865265

Oltrogge, M., Derr, E., Stransky, C., Acar, Y., Fahl, S., Rossow, C., Pellegrino, G., Bugiel, S., & Backes, M. (2018). The Rise of the Citizen Developer: Assessing the Security

Impact of Online App Generators. *Proceedings - IEEE Symposium on Security and Privacy*, *2018-May*, 634–647. https://doi.org/10.1109/SP.2018.00005

Project Management Institute. (2021). *Citizen development : The handbook for Creators and Change Makers*.

Sahinaslan, E., Sahinaslan, O., & Sabancioglu, M. (2021). Low-code application platform in meeting increasing software demands Quickly: SetXRM. *AIP Conference Proceedings*, *2334*(March). https://doi.org/10.1063/5.0042213

Salgueiro, R. U. B. (2021). *The impact of Microsoft Power Platform in streamlining end-to-end business solutions-Internship Report at Microsoft Portugal, Specialist Team Unit*.

Shih, K., Chang, P., & Myers, B. A. (2017). Gneiss: Spreadsheet Programming Using Structured Web Service Data. *Journal of Visual Languages & Computing*. https://www.elsevier.com/open-access/userlicense/1.0/

Silva, C., Vieira, J., Campos, J., Couto, R., & Ribeiro, A. (2020). Development and Validation of a Descriptive Cognitive Model for Predicting Usability Issues in a Low-Code Development Platform. *Human Factors*. https://doi.org/0018720820920429

Singh, P. (2021). *The Good And The Bad Of Citizen Development*. Forbes. https://www.forbes.com/sites/forbestechcouncil/2021/07/07/the-good-and-the-bad-of-citizen-development/?sh=541567fe5d6a

Thacker, D., Berardi, V., Kaur, V., & Blundell, G. (2021). Business Students as Citizen Developers: Assessing Technological Self-Conception and Readiness. *Information Systems Education Journal (ISEDJ)*, *19*(October), 18.

Tisi, M., Mottu, J. M., Kolovos, D. S., de Lara, J., Guerra, E., Di Ruscio, D., Pierantonio, A., & Wimmer, M. (2019). Lowcomote: Training the next generation of experts in scalable low-code engineering platforms. *CEUR Workshop Proceedings*, *2405*.

Vincent, P., Iijima, K., Driver, M., Wong, J., & Natis, Y. (2019). *Magic Quadrant for Enterprise Low-Code Application Platforms*. https://www.gartner.com/doc/reprints?id=1-1XQ92DO5&ct=191105&st=sb

Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. *IFAC-PapersOnLine*, *52*(10), 376–381. https://doi.org/10.1016/j.ifacol.2019.10.060

Wong, J., Driver M, & Ray S. (2019). *The Future of Apps Must Include Citizen Development*. https://www.gartner.com/en/documents/3970067

Woo, M. (2020). The Rise of No/Low Code Software Development—No Experience Needed? *Engineering*, *6*(9), 960–961. https://doi.org/10.1016/j.eng.2020.07.007

Yan, Z. (2019). *The Impacts of Low/No-Code Development on Digital Transformation and Software Development*.

Yang, C., Pan, S., Li, R., Liu, Y., & Peng, L. (2019). A coding-free software framework for developing lightweight web data management systems. *Applied Sciences (Switzerland)*, *10*(3). https://doi.org/10.3390/app10030865

Yu, J. (2018). Research Process on Software Development Model. *IOP Conference Series: Materials Science and Engineering*, *394*(3). https://doi.org/10.1088/1757-899X/394/3/032045