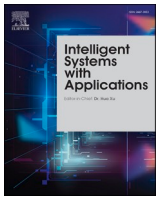
Intelligent Systems with Applications 20 (2023) 200289

Contents lists available at ScienceDirect

Intelligent Systems with Applications

journal homepage: www.journals.elsevier.com/intelligent-systems-with-applications



# Combining low-code development with ChatGPT to novel no-code approaches: A focus-group study

José Martins [a,b], Frederico Branco [a,c], Henrique Mamede [a,d,*]

[a] INESC TEC—Institute for Systems and Computer Engineering, Technology and Science, 4200-465 Porto, Portugal
[b] Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300-253 Bragança, Portugal
[c] Department of Engineering, School of Sciences and Technology, Universidade de Trás-os-Montes e Alto Douro, 5000-801 Vila Real, Portugal
[d] CEG-UAb, Universidade Aberta, Rua da Escola Politécnica, 147, 1269-001 Lisboa, Portugal

ARTICLE INFO

*Keywords:*
Low-code
No-code
Artificial intelligence
Software models
ChatGPT
LLM

ABSTRACT

Low-code tools are a trend in software development for business solutions due to their agility and ease of use. There are a certain number of vendors with such solutions. Still, in most Western countries, there is a clear need for the existence of greater quantities of certified and experienced professionals to work with those tools. This means that companies with more resources can attract and maintain those professionals, whilst other smaller organizations must rely on an endless search for this scarce resource. We will present and validate a model designed to transform ChatGPT into a low-code developer, addressing the demand for a more skilled human resource solution. This innovative tool underwent rigorous validation via a focus group study, engaging a panel of highly experienced experts. Their invaluable insights and feedback on the proposed model were systematically gathered and meticulously analysed.

## 1. Introduction

A new paradigm called Low-code Systems Development (LCSD) is shifting software applications development into something with far less manual coding by using visual programming approaches aided by graphical user interfaces and model-driven designs, typically supported by technological platforms (Al Alamin et al., 2021). These (low-code) platforms are a type of software development tool which is used for software design and development. It is based on using a graphical user interface (GUI) and pre-built elements and components, such as user interfaces, business objects, data objects (e.g., tables), and other components. The GUI and the pre-built components can easily be combined into complete solutions by requiring users to provide input in data fields, user interface elements, and business logic (Chen et al., 2022).

Furthermore, LCSD not only offers a platform for developing solutions that are typically easier to understand but also provides for the reuse of components from other solutions and the extension of the pre-built components functionality by allowing its customization (Bock & Frank, 2021). With such platforms, developers are writing less code and concentrating on the goal of the system being developed

Nevertheless, and as argued by existing literature, the use of LCSD platforms can also provide unpleasant experiences that, from the developers' perspectives, might negatively impact the potential benefits it would arise. These issues are directly related to possible work constraints, limited freedom and creativity, inadequate documentation and overview, and poor and unsafe teamwork capabilities (Beranic et al., 2020; Conchúir et al., 2009; Gao, 2022).

As established by authors such as Blanchard (2013) and Breaux and Moritz (2021), the globally recognized shortage of qualified developers, a problem transversal to all technological fields, not only poses new challenges to the growing need for new software applications in an increasingly interconnected World but also tends to weaken significantly the negative perspectives that might arise towards LCSDs.

As the perennial issue with software development continues to be that its endeavours typically fail due to inefficiency issues from the development teams (Bock & Frank, 2021; Casadei et al., 2019), independent researchers, developers, and organizations have been collaborating towards new Large Language Models (LLM) base approaches. The most famous result of this work is, without a doubt, OpenAI.[1] "ChatGPT" platform (Wang et al., 2023). This platform's ability to participate in free-form discourse and continuously and extensively collect context allows users a more natural interaction and a more flexible behaviour

* Corresponding author.
*E-mail address:* jose.mamede@uab.pt (H. Mamede).
[1] OpenAI: https://openai.com/.

https://doi.org/10.1016/j.iswa.2023.200289
Received 11 July 2023; Received in revised form 19 September 2023; Accepted 9 October 2023
Available online 12 October 2023

correction. Despite having flaws, ChatGPT's enhanced capabilities open the door to various applications in multiple industries (Dwivedi et al., 2023).

ChatGPT can, amongst other things, answer questions, produce content, restyle text, develop code and debug it, take tests, modify data, explain and tutor. Hence, drawing on the convergence of those topics, we established a proposition that would serve as a guideline for the research endeavour inherent to this article: "Is there any model where ChatGPT can overcome the lack of code developers, specifically using low-code platforms?". Aiming to respond to this question, we propose an effective and efficient model for combining ChatGPTs' abilities with human code developers.

### 1.1. Methodological approach

From a conceptual perspective, this research draws on a preliminary identification of a potential problem for those involved in the software applications development activity, on the recognition of the current relevancy of that same issue, and on the need to not only propose a solution to it but also to assess its adequacy and validity. Thus, drawing on the existing literature, namely similar research projects, it was established that this research would follow Hevner et al. (2008) "Design Science Research" (DSR) methodology.

The DSR aims to help researchers address identified organizational problems by designing and evaluating Information Technology (IT) artefacts.

As shown in Fig. 1, the six steps established in DSR will be followed by the research team that will ensure the execution of the following activities:

- Step 1 - The problem is identified, and the research background will set the stage for the justification of the research question;
- Step 2 – The objectives and requirements for a solution are defined, knowing that aim at proposing a novel model for successfully using ChatGPT alongside LCSD platforms and doing so with as little human intervention as possible;
- Step 3 – The proposed model is presented in a detailed manner;
- Step 4 – A demonstration of the proposed model is achieved, thus allowing us to perceive how it copes with the particular context in which we would like to test it.
- Step 5 – For evaluation purposes, a focus group will be used;
- Step 6 – This last step consists of disseminating the achieved knowledge.

Considering the scope of our research and the potential complexity inherent to the proposed model, it was established that a Focus Group would best fit our purpose of ensuring a proper assessment of the referred artefact. A Focus Group is a technique that aims to collect data and may be used at different moments of the research process through group interaction on a topic presented by the researcher (Gonçalves et al., 2016). As reasoned by Guest et al. (2017) and Santos and Boticario (2015), the focus group definition includes three essential components: 1) a research method aimed at data collection; 2) locating the interaction in the group discussion as the source of data; and 3) recognizing the active role of the researcher in stimulating the group discussion for data collection purposes.

According to the analysed literature (Camarinha-Matos et al., 2019; Mather & Cummings, 2017), the Focus Group is one of the instruments to be implemented when aiming at enhancing existing knowledge by using cross-disciplinary perspectives and synergies from experts in the field of study. This instrument allows information and data collection, producing compelling expertise and insights. Specifically, the synergy and dynamism generated within homogeneous collectives often reveal norms and normative assumptions.

Thus, the Focus Group technique will be applied in "step 5" (Fig. 1) and encompass the pre-defined and sequential activities outlined in Fig. 2.

This paper is organized as follows. Section 1 describes both the context for the research and the methodological approach that was drawn and supported the entire research. Section 2 describes the theoretical background analysis that upheld the research. This is followed by a third section, where the proposed artefact is contextualized and detailed. Section 4 holds the characterization of the Focus Group developed to validate all the perspectives and arguments inherent to the proposed model. Sections 5 and 6 present the achieved results analysis and our research's overall considerations and conclusions.

## 2. Theoretical context analysis

This section discusses the main topics inherent to our research project's theoretical background, thus ensuring that the proposed artefact aligns with existing scientific knowledge.

### 2.1. Low-code development approaches and life-cycle

Despite only being commercially coined in 2014 by Forrester (Richardson et al., 2014), the first peer-reviewed studies on LCSD did not appear until 2018. Since then, there has been a considerable increase in articles published on LCSD-related topics.

LCSD is a semi-automated visual approach to software development. The core idea is that by using LCSD, a software engineer can abstract and semi-automate every aspect of a software system development life-cycle while also accelerating the delivery of the numerous applicational modules that compose it (Bucaioni et al., 2022; Sanchis et al., 2019; Sarkar, 2022).

LCSD platforms include application builders or makers that allow users (developers) to *drag-n-drop* components, diagrams and forms to assemble functional versions of a given application, which are run within a platform engine, often as a cloud service (Juhas et al., 2022).

From a conceptual point of view, the low-code development life cycle is a software development process that uses low-code development tools and platforms to create more efficient applications that need fewer lines of code than standard software development approaches. In general, the following steps are often included in the low-code development life cycle (Luo et al., 2021): 1) Gathering needs: In this phase, information on the applications' functionality, user experience, and business requirements is gathered; 2) Design: The program's architecture, user interface, and other design features are developed during this phase. Wireframing and prototyping may be included; 3) Development: This phase is dedicated to the actual development of the application. Low-code platforms often include visual editors and drag-and-drop tools to aid development; 4) Testing: The application is tested for bugs, problems, and usability concerns. Low-code systems frequently provide automated testing tools that ease this process; 5) Deployment: Once tested and approved, the application is deployed to a production environment.

Low-code platforms frequently include monitoring and maintenance features, such as automatic error reporting and deployment rollback. The low-code development life cycle is intended to be faster and more efficient than traditional software development approaches, allowing enterprises to build and deploy systems with fewer resources and less time.

Despite several methodological approaches focused on developing software solutions using a low-code platform (Al Alamin et al., 2021; Elshan et al., 2023; Gottschalk et al., 2022), a significant part tend to be restricted to the platform itself and its functionalities (Krishnaraj et al., 2022). Drawing on the argument that one needs a broad methodological approach that can support the development processes, regardless of the chosen low-code platform, for this research, the research team decided that the method presented by Alamin et (2021) would be the most adequate hence it was the one that we have chosen to support the remaining of the research.

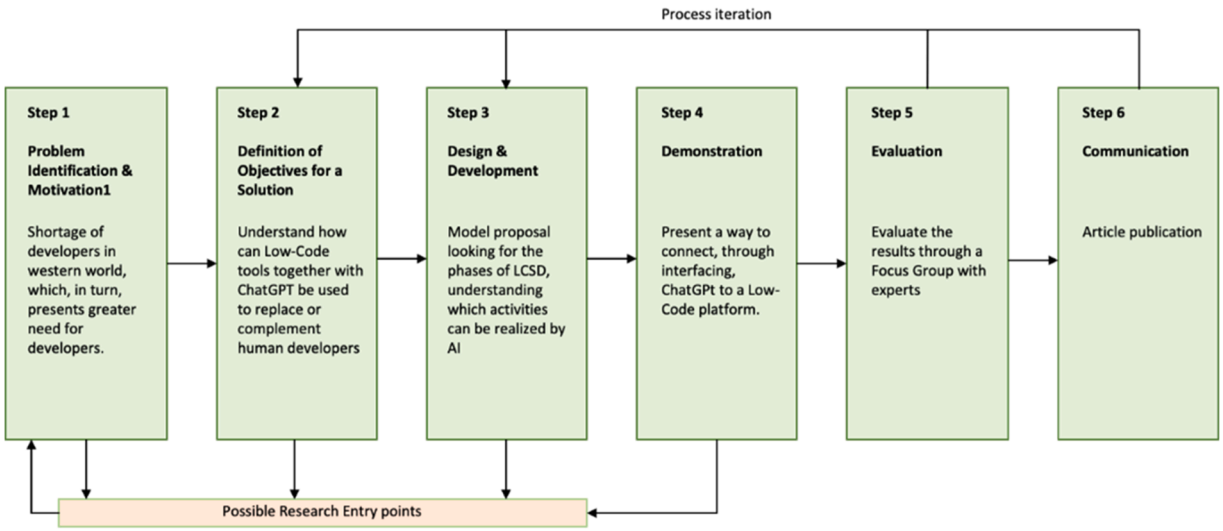



**Fig. 1.** DSR methodology steps. Adapted from Hevner et al. (2008). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

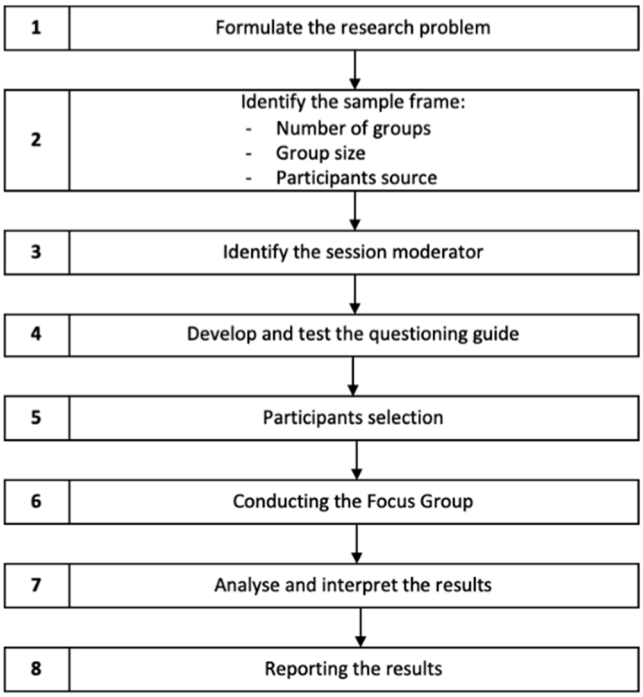



**Fig. 2.** Steps of the performed Focus Group. Adapted from Tremblay et al. (2010).

As stated by Alamin et (2021), there are several stages for developing software solutions using a low-code platform (Fig. 3). This is the life-cycle approach we will use as a basis for our proposal.

In the first stage, data modelling, the developer will look at the requirements already collected by the business analyst or identified in a meeting with the final users and confirmed by them. Those requirements are then transformed into the data model, information for application flow through functionalities and later user interface design. This will take the process to the second stage, of developing the user interface, based on the data model already defined. The third stage is business logic implementation, where the rules for each information entity and the application flows are implemented. The next stage is integrating external services and the development interfaces for existing applications and services. The fifth stage concerns testing what has been developed and deployed and training the user on the new application. During regular operation, collecting user feedback for future improvements or additional features that could be needed through time updating the application is essential.

### 2.2. Market situation for professional low-code developers

In the United States, it is said that around 40 million technical jobs will go unfulfilled due to a lack of skilled talent in 2022, the most significant part composed of code developers (Philips, 2022).

The developer shortage will be around for a while, with the most technologically advanced countries expecting to lack over 1 million developers by 2026 (Beres, 2022). While some authors argue that low-code tools can help this shortage of code developers, others say that the problem could be in obsolete recruitment processes and the need to be more agile for this type of skilled recruitment (Sloyan, 2021).

Although the significant differences in the numbers presented by different entities, as stated in the previous references, one thing can be acknowledged. There needs to be more code developers in the Western world.

### 2.3. ChatGPT—Current perspectives

In November 2022, Open AI released an artificial intelligence (AI) chatbot named ChatGPT. It is based on the OpenAI GPT-3 family of large

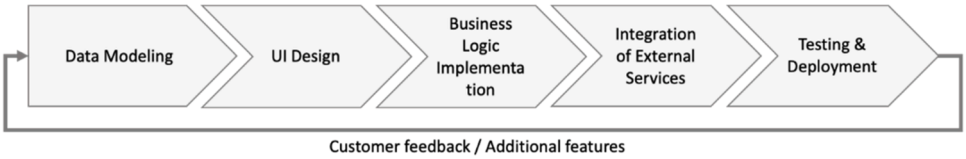



**Fig. 3.** Low-code stages of development. Adapted from Al Alamin et al. (2021).

language models and has been utterly improved through supervised and reinforcement learning strategies.

As Aljanabi (2023) argues in his research, merging ChatGPT with features and services from other technologies will allow for the development of intelligent and conversational AI systems that might fundamentally alter how humans interact with technology.

These modern language processing models are the result of the tremendous advancement of AI in recent years (Rizou et al., 2023). With enormous data and complex language models (such as ChatGPT), many language-related tasks, including text production, question answering, and even poetry composing, are now possible (Köbis & Mossink, 2021). It is a popular choice for many applications, including customer service and content creation, because of its excellent performance (Brown et al., 2020; Jafarinejad, 2023).

ChatGPT and other conversational AI models offer significant and far-reaching potential benefits, notwithstanding the ethical concerns. For instance, they may provide round-the-clock help in the customer service sector and improve the whole client experience, reducing wait times and improving the quality of interactions (Carlbring et al., 2023). They may also be used in marketing and content production to create high-quality outcomes, freeing human workers to focus on more creative and strategic tasks (Gursoy et al., 2023). As pointed by Bidochko (2023), other benefits must be noted like increased efficiency and productivity, enhanced customer experience, cost savings and increased innovation. Other authors also highlight aspects such as the lack of size and technological competence needed for solutions that require advanced coding skills, that can be overcomed by using AI (Sundberg & Holmström, 2023).

These conversational AI models change how we interact with technology, altering how we communicate and access information (Jafarinejad, 2023; S. Wang et al., 2022). Chatbots, for example, may promptly and individually answer users' enquiries without human intervention (Mohamad Suhaili et al., 2021). This might significantly affect industries like healthcare, where chatbots may counsel people on their ailments, drugs, and other health-related problems (Xu et al., 2021).

Hence, as stated above, the Chat-GPT model is regarded as a cutting-edge language model capable of carrying out various language activities, such as answering queries, translating across languages, and producing original material.

In their research, Sobania et al. (2023) have demonstrated what to expect from ChatGPT regarding mistake and defect correction in human-written code, along with other automated program repair (APR) systems like CoCoNut and Codex. A recently released deep learning (DL) based chat system called ChatGPT may also offer advice for fixing flawed source code. However, it is still being determined how good these proposals are now. They discovered that ChatGPT performs similarly to Codex and specialized DL-based APR on a standard benchmark. It works far better than conventional APR techniques. Using ChatGPT's conversation feature and sending the system a follow-up request with further details about the problem improves speed even more.

However, as stated by Guo et al. (2023), we are interested in seeing how comparable ChatGPT is to actual specialists. In contrast to earlier large language models (LLM) like GPT-3 (Brown et al., 2020), which frequently struggles to respond to human inquiries effectively, InstructGPT (Ouyang et al., 2022) and the more potent ChatGPT have greatly improved in interactions with people. As a result, ChatGPT has considerable potential to become a helpful tool for general or corporate consulting (Burger et al., 2023; Pan & Nishant, 2023).

According to Jalil et al. (2023), when ChatGPT is just confident rather than utterly sure in its response, it is twice as likely to be correct. This suggests that ChatGPT needs a better grasp of its responses' accuracy. In other words, the answers provided by ChatGPT will need to be corrected, at least for the time being.

Developers now have access to cutting-edge language instead of merely chat, thanks to the ChatGPT model being made available through API. Users of the ChatGPT API may anticipate ongoing model upgrades and the choice of dedicated capacity for more control over the models.

Unstructured text is typically consumed by GPT models and is supplied to the model as a series of "tokens". Instead, ChatGPT models ingest a series of messages together with their associated information. The model uses a brand-new format called Chat Markup Language (ChatML) as its raw format (Brockman et al., 2023).

## 3. No-code development—The new future

Even though low-code and no-code platforms can help businesses build and modernize more apps, we are still coding microservices, designing customer-facing applications, and developing machine-learning capabilities (Johnsson & Magnusson, 2020). While little or no code will not replace traditional developers and software engineers, ChatGPT will provide critical tools to eliminate repetitive tasks and enhance app development time to market (Chen et al., 2022). ChatGPT can generate boilerplate or recommended example code for problems considerably faster than any developer can write and test code from the ground up. AI will assist developers in speeding up repetitive judgements that engineers must make, such as general language enquiries (Sundberg & Holmström, 2023). Our capacity to access information more rapidly will improve, as will our productivity.

Some examples of input that an AI tool like ChatGPT can generate for low-code platforms (Dwivedi et al., 2023; Esposito et al., 2023):

- Data fields: A list of data fields that may be utilized to store data in the application can be generated by it. These fields can have text, numeric, date, dropdown, checkbox, and radio buttons, amongst others.
- User interface elements: It can produce user interface components, including buttons, menus, tabs, and models. These components may be used to interact with the application's features and traverse the interface.
- Business logic: It may produce conditions and rules that describe how an application should behave. For instance, it can create logic to conduct computations, verify user input, and start activities depending on specific occurrences developing code snippets, if needed, and automating repetitive coding tasks.
- Integration with other systems: It can produce data for the application's integration with other systems, including databases, APIs, and outside services. Data mapping rules, endpoint URLs, and login credentials are examples of this input.
- Debugging and Error Correction: ChatGPT can be trained on error messages and suggest possible solutions for code errors. This can help speed up the debugging process and improve error correction accuracy. It can also write test cases, assisting the developers in testing implemented functionalities.
- Documentation Generation: ChatGPT can generate documentation for code by processing natural language input and generating descriptions of code functions and parameters. This can save time and improve the readability of code documentation.

Overall, using ChatGPT may produce a wide range of input for low-code platforms to assist users in rapidly creating unique applications, even without being able to replace the creativity and problem-solving abilities of human coders. Developers still need to be involved in the development process to ensure that the code generated by ChatGPT is high quality and meets the project's specific requirements (Liu et al., 2023).

In the current development life-cycle, as Fig. 4 illustrates, human developers are involved in all application development steps using LCSD platforms, leading and performing all needed activities without regard for their complexity. In the current model, we consider the existence of humans as the only actors with intervention in the whole process.

Replacing human-dependant activities with tasks executed by

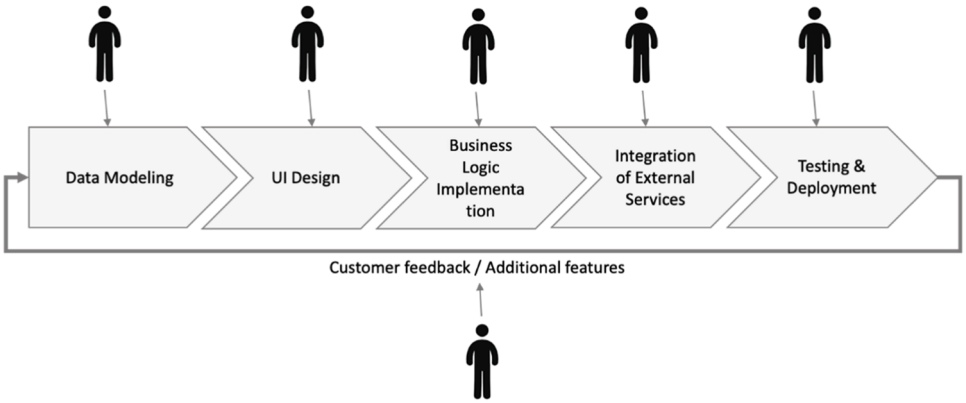



**Fig. 4.** LCSD life-cycle and human intervention. Adapted from Al Alamin et al. (2021).

ChaGPT is, according to existing literature, the most feasible manner to address the main issues associated with application development: low satisfaction of developers with LCSD platforms and shortage of developers to be hired by companies needing to develop or maintain advanced solutions. Hence, considering this critical scope, one can propose: "Which activities can AI perform, and what remains to be done by the human coder?". This is the core of our proposed innovative model, illustrated in Fig. 5.

Unfortunately, neither ChatGPT can handle all these situations perfectly nor the LCSD platforms are prepared to work following the proposed model. Furthermore, we must consider that ChatGPT is not a replacement for human coders and cannot fully automate the development process. Although ChatGPT is not designed to develop applications directly or interface with LCSD platforms, it can provide guidance, answer questions related to the development process, and offer suggestions on approaching specific development tasks, thus allowing for quicker and more efficient development cycles.

Hence, as an intermediate step, we propose that ChatGPT adopts the leading role in the development activities but that code developers always validate its outcomes to ensure the quality and suitability of what code is AI-generated. This intermediate model is illustrated in Fig. 6.

According to the proposed model, ChatGPT will be developing activities in the stages of UI design, Business Logic Implementation, and Integration of External Services, typically through the configuration of connectors and also in the Testing and Deployment phases. Since those activities cannot be entirely relied upon by ChatGPT, there is a validation activity to be performed by the human coder, changing or completing whatever needs to be put by the objective of developing the application. The exciting part is that ChatGPT can do the more time-consuming work while the human coder can perform more valuable activities. In this intermediate model, AI can be seen as a companion for the code developer, performing all of the most time-consuming tasks.

In the Data modelling stage, relying only on human coders is essential as ChatGPT-generated models tend to strictly and mindlessly follow Codd's rules, which only sometimes allow for the achievement of a valid data model. An AI language model can generate text describing and explaining data models without directly constructing or developing them. Data modelling is an activity where the developer creates a structured representation of data and connections between various items. Typically, the process begins with defining data items, properties, and connections, then structuring them into a model using standardized notation such as ER or UML diagrams. Data analysts or data architects with particular expertise and training in data modelling are often responsible for this (Bogdanova & Snoeck, 2019).

The connection of ChatGPT to the LCSD platform has to be achieved through OpenAI APIs. The platform vendors must accomplish this before this model can be implemented.

As established form existing literature, the use of AI tools, such as ChatGPT, as active supports to the low-code software development lifecycle is yet a very recent achievement, with little to none examples on how to methodologically merge the features and benefits of the referred tools with the software development based on the low-code principals and standards. Hence, by proposing – in a detailed manner – the combination of ChatGPT (or other AI tool) combined with low-code platforms, as active supporters to the entire software development lifecycle, the artefact we hereby present ought to be considered both innovative and disruptive.

## 4. Focus group study

This section describes the validation stage of the proposed artefact, corresponding to the fourth step of DSR. In order to perform this

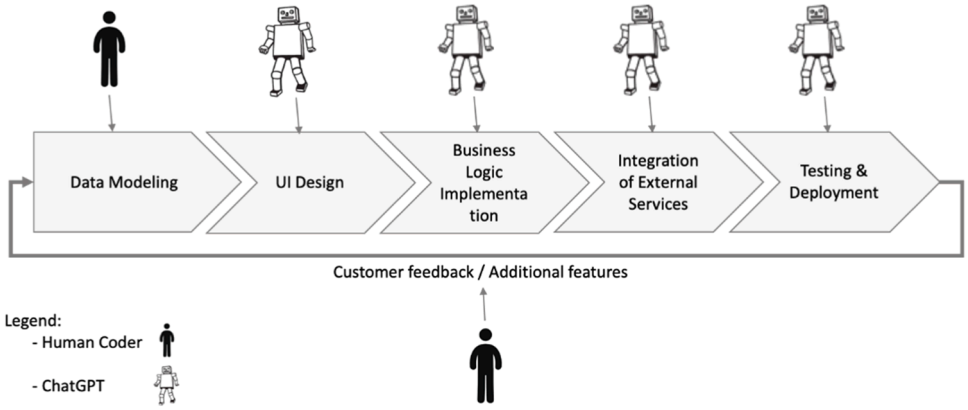



**Fig. 5.** LCSD life-cycle with human and ChatGPT intervention model proposal.

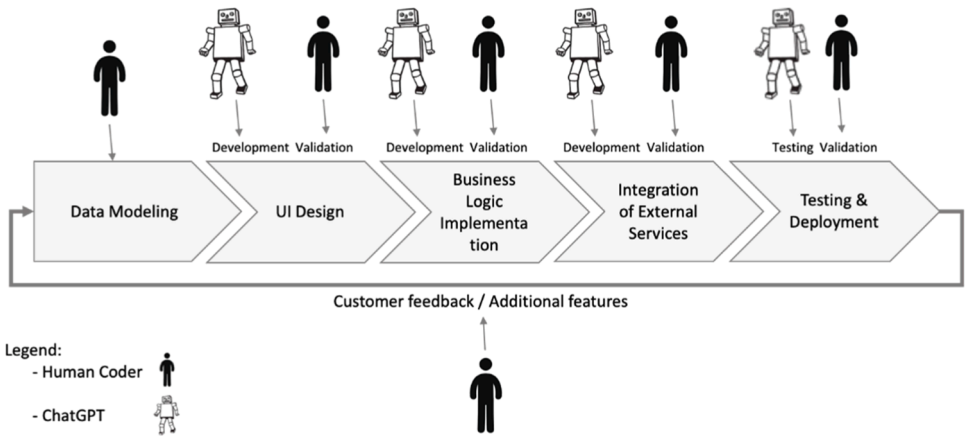



**Fig. 6.** LCSD life-cycle with human and ChatGPT intervention adjusted.

assessment of our proposal, a focus group has been operationalized, where the proposed model was demonstrated to specialists in domains related to low-code platforms and software engineering.

### 4.1. Planning

Since the research problem was already defined, we had to determine the content and the Focus Group. Considering the complexity of the task, it was decided to execute two editions of the proposed Focus Group: the first to serve as a control group, using scholars from renowned Portuguese Universities, to test the execution times and adequacy of the content. After this test session, a second session, now encompassing the abovementioned experts, was planned.

Demonstrating the value of the artefact's creation was also addressed while designing the Focus Group session, resulting in a combination of both exploratory and confirmatory perspectives (Stewart & Shamdasani, 1998).

In this scenario, six participants were chosen to represent the perspectives inherent to developing software solutions with low-code platforms: higher education professors, researchers, end-users, developers, and platform developers (see Table 4). The authors controlled moderation and closely adhered to the instructions (Tremblay et al., 2010).

#### 4.1.1. Questions and attributes to evaluate

The questions raised throughout the session should always be related to the research issue (Kidd & Parshall, 2000). However, the proposed artefact and its qualities entail the inclusion of other viewpoints connected to its capacity to address the defined problem and deliver an approach that is both helpful for organizations and of high quality and successful in its application.

The questions that were posed to the focus group experts were asked in a particular order and structure, thus ensuring that:

- No questions were asked directly to a participant or about a specific area of expertise;
- The required ideas or constructs that support the traits or questions stated were delivered to participants while keeping an acceptable amount of knowledge for discussion periods;
- There were opportunities for participants to share and contribute their experiences and generate recommendations for enhancing the artefact based on their understanding of the topics under discussion.

As argued by Morgan (1996), qualitative assessment criteria were established to target the conversation and provide arguments and acknowledgements that could be compared between iterations (Table 1).

Using the abovementioned attributes (Table 1), we captured participants' agreement towards their relationship with the proposed artefact. All attributes were measured using a scale from 0 to 100 %, where 0 % represented "completely disagree" and 100 % represented "totally agree".

**Table 1**
Attributes for qualitative assessment. Adapted from Hennink (2014); Krueger and Casey (2002).

| Attribute | Meaning |
|---|---|
| Clarity | Intelligible and transparently expressed |
| Complexity | Meaningful concepts |
| Feasibility | Potential to be implemented and used |
| Maintainability | Able to be maintained, improved and updated |
| Completeness | Covers the essential issues of developing low-code platforms |
| Consistency | A conceptual model that identifies the components and relationships |
| Cohesion | Components and activities work together to implement solutions on low-code platforms. |
| Scalability | Open, able to grow evenly and support further orientation |
| Adaptability | Potential to be applied to all types and sizes of solutions |
| Prescriptive | Provides insight into what needs to be addressed in each component or activity |
| Flexibility | It makes it possible to add guidelines for new types of development and pattern use |
| Management Support | Allows management of activities |

The questions to be posed, the creation of the session guide and questioning structure, and the defined technique and tactics to be used were drawn from Krueger and Casey (2002). The final list of questions is presented in Table 2.

### 4.2. Test session

As previously mentioned, the first session was meant to serve as a control group, using scholars from renowned Portuguese Universities to test the execution times and adequacy of the content. All the participants in the test session were Professors from Portuguese Universities.

Table 3 holds the participant's profile description.

With this session, both the execution times and content adequacy were tested, and, considering the achieved results, no adjustments to the established protocol were needed.

### 4.3. Participants analysis

In order to ensure the maximum validity and knowledge-generation capabilities from the performed Focus Group, a group of highly-specialized and senior participants has been gathered. The participant's area of activity and experience are detailed in Table 3.

**Table 2**
List of questions for focus group.

| Question # | Type | Description |
|---|---|---|
| G1 | General | What is your experience, empirical or conceptual, with low-code platforms? |
| G2 | General | To what extent are artificial intelligence technologies currently incorporated into the software development process? |
| G3 | General | Are you familiar with chat agents like ChatGPT and their possible applications in software development? |
| G4 | General | In your opinion, what are the benefits of integrating chat agents like ChatGPT into low-code platforms? |
| G5 | General | Do you consider that the integration of conversational agents in low-code platforms would improve the efficiency of software development? |
| G6 | General | Do you identify limitations or constraints in integrating AI technologies like ChatGPT into your software development process? |
| G7 | General | Describe how integrating conversational agents into low-code platforms can impact the dynamics of software development teams. |
| G8 | General | What features would you like to see implemented with a chat agent integrated into a low-code platform? |
| G9 | General | What training is needed for software development teams to use chat agents like ChatGPT on low-code platforms? |
| S1 | Specific | To what extent can ChatGPT help specify the data model in software projects based on a low-code platform? |
| S2 | Specific | How can ChatGPT help design user interfaces in software projects based on a low-code platform? |
| S3 | Specific | When implementing business logic in low-code projects, how can ChatGPT be a differentiating agent? |
| S4 | Specific | What are the benefits and constraints of using ChatGPT as functionality to support integrating external services in low-code platforms? |
| S5 | Specific | How can ChatGPT be a differentiating element in the low-code application testing and deployment phase? |
| G10 | General | We would like to have everyone's last comment as a closing statement. |

**Table 3**
Test session specialists' profile description.

| Participant | Area of activity | Experience |
|---|---|---|
| P_A | Teacher/ Researcher | Assistant professor with twenty years of teaching and scientific research in the field of Computer Science and Engineering. |
| P_B | Teacher/ Researcher | Associate professor with twenty-two years of teaching and scientific research in the field of Computer Science and Engineering. |
| P_C | Teacher/ Researcher | Assistant professor with twelve years of teaching and scientific research in the field of Computer Science and Engineering. |
| P_D | Teacher/ Researcher | Assistant professor with eight years of teaching and scientific research in the field of Computer Science and Engineering. |
| P_E | Teacher/ Researcher | Associate professor with nineteen years of teaching and scientific research in the field of Computer Science and Engineering. |

A brief analysis of the abovementioned profiles demonstrates that the focus group participants included both researchers and practitioners with at least three different perspectives: platform vendor, developer, and client-side developer.

The diverse collection of experts brought together triggered the active formation of ideas, the diversity of contributions, and the development of in-depth conversation, thus ensuring a proper debate and knowledge generation.

### 4.4. Focus group execution

The Focus Group session was held on the 5th of May 2023, with a total duration of 138 min, through a Web conference system. Despite initially planning to last 120 min, the focus group was so active that it was extended for an extra 18 min to ensure that all opinions, perspectives and agreement points were discussed in detail.

In what concerns the operationalization of the Focus Group, The study was supported by the following (sequential) stages:

1. Initial presentation of the session moderator, introduction to the study and brief presentation of each participating expert;
2. Presentation of the session technique and data collection;
3. Presentation of the proposed development model;
4. Presentation of discussion questions and model attributes defined for evaluation;
5. Discussion;
6. Session conclusion and synthesis;

In order to prevent the existence of previous bias towards the proposed artefact, no information or details were previously made accessible to the focus group participants.

After the initial presentation stages, the moderator gave access to the session script and information on the proposed artefact and its components, thus revealing how it addresses the problems inherent to the software development topic. In order to further trigger the discussion, the focus group participants were encouraged to position themselves as users of the artefact in an organization by simulating its usage. They were asked to compare the outcomes of using the item in an organization against the results of not utilizing it or some other comparable tool or approach. Following a moment when the experts were asked to analyse the proposed artefact summary description, they were also invited to assess the model's utility, effectiveness, and quality through a conversation guided by the scheduled questions and an evaluation of the referred model against the established quality attributes.

### 4.5. Focus group results

After presenting the model to the participants and introducing the questions and attributes, the participants discussed it openly. This discussion moment was always closely monitored by the moderator, who recorded in detail and continuously all the arguments being presented and discussed.

Once the focus group was over, the research team analysed all the information gathered, as well as all the arguments and perspectives that were reached and considered consensual. Analysing the achieved results allowed us to perceive a set of contributions and perspectives that represented a detailed assessment of the proposed artefact validity and overall potential value (Table 4) and a precious input for further

**Table 4**
Focus group specialists profile description.

| Participant | Area of activity | Experience |
|---|---|---|
| P1 | Teacher/ Researcher | Thirty years of teaching and scientific research. Currently Full Professor in the area of Computer Science and Engineering. |
| P2 | Practitioner/ Consulting | Thirty years of experience in different roles in Solutions Development, with a focus on low-code tools in the last 18 years. Currently, Director of an International Consulting Company. |
| P3 | Practitioner/ Client Side | Twenty-five years of experience in different roles in Solutions Development, with a focus on low-code tools in the last 15 years. Currently, Software Development Director in a Portuguese Company. |
| P4 | Practitioner/ Vendor | Thirty-five years of experience as a software developer and database administrator. She is currently working on a low-code vendor. |
| P5 | Researcher | Researcher on Software Engineering (research directed to obtain the PhD) with 15 years of experience as a full-stack code developer |

improvements (Table 5).

Table 6

Drawing on Hennink (2014) and Krueger and Casey (2002), the set of attributes for qualitative assessment previously presented (Table 1) was also addressed by the focus group and the results of this interaction in perceivable in Table 7. Most attributes obtained a 100 % evaluation, thus indicating that all focus group participants confirmed its description. From the referred discussion, it was also possible to highlight a set of contributions and different perspectives that further contribute to the overall knowledge generation factor of the research.

Considering the potential inherent to the contributions and perspectives achieved with the focus group, these will serve as the basis for future improvements to the proposed model.

## 5. Focus group achieved perceptions and suggestions

As argued by Tian et al. (2023), we are currently experiencing an increase in the adoption of both artificial intelligence techniques and large-scale language models, such as ChatGPT, as both these tools have an immense potential for addressing tasks such as problem-solving, code generation, code repairing and for both programming and software engineering-related tasks.

Furthermore, as established by Cait et al. (2023), LLMs also represent a handy tool for low-code programming operations as low-code platforms provide visual interfaces where developers can implement developing AI pipelines that AI-generated instructions can fully automate.

Although the performed focus-group participants all agreed with the arguments above, it was possible to acknowledge a set of individual perceptions and suggestions that represented both a view on the global topic surrounding the research and an overall contribution to the advancement of the proposed LCSD life cycle with human and ChatGPT intervention model.

As established by the focus group experts, despite at this point-in-time it is not an easy task to be performed, it is expected that in a near future, ChatGPT will be able to perform software modelling with only minimal human-intervention, thus becoming an aid for both software engineers and software developers. This argument is inline with Combemale et al. (2023), according to whom the task of software modelling will be completely transformed with the advent of LLMS and particularly with ChatGPT and other similar tools, as these will have the ability to propose the entire set of modelling artifacts inherent to a software solutions just from analysing a set of (well established) requirements, and the human interaction will only serve for approval purposes.

Furthermore, in the same near future mentioned before, ChatGPT is also expected to incorporate features that will allow it to actively identify and formally establish software requirements by analysing textual descriptions of the problems that need to be addressed (Ahmad et al., 2023; Li et al., 2022) or by ensuring active dialogues with the process stakeholders (Tiwari et al., 2023). Drawing on this premise, and on the difficulty inherent to interact in a textual manner with ChatGPT, the focus group experts pose that a future step could be the development of conversational features that would allow it to simply interview the business stakeholders and from this interaction to extract a detailed description of their needs and inherently propose the requirements for the software solution that could address those same needs.

After carefully considering the future potential of ChatGPT (and other LLMs) it was also proposed by the focus group experts that the inherent ability to easily – and with just little human intervention - develop webpages, forms, and other components of a software solution (Tian et al., 2023) and doing so by following existing best-practices and standards, could be the trigger needed for not only speeding up the development phases duration and resource consumption, but also to ensure that the developed software solutions would all follow the industry best practices in terms of architecture, data model manipulation,

**Table 5**
Topics discussed by the focus group and additional remarks that have arised.

| Topic | Remark |
|---|---|
| The Group discussed that ChatGPT could assist in the data modelling phase. Using the requirements collected by the human developer, ChatGPT can identify the main concepts and their attributes, from where it can generate a draft model. | Inclusion of ChatGPT in the Data Modelling phase. |
| Current low-code platforms have already demonstrated to have some AI technology embedded, allowing them to suggest more appropriate development patterns. Thus, the result produced by the developer can be more consistent and scalable, according to those suggestions provided by the AI. However, the final decision is always made by the developers. Other forms of AI are only experimental and not used in a structured manner for development purposes. | It goes in the direction of our research, confirming our interest in it. |
| ChatGPT processes text and answers - A text-based conversational machine expected to incorporate voice in a very near time, low-code (or no-code) coding can have significantly higher levels of abstraction. However, we will always have to instruct the machine on what needs to be built, and this is where it might make more sense to use ChatGPT to improve how it interacts with the low-code platform itself. | Confirmation of our research. |
| Integration of ChatGPT in the development process can improve the detection of problematic codding patterns, thus being able to perceive the solution as a whole. It enhances the ability to improve coding efficiency without removing the human developer from the development process. | Confirmation of our research. |
| The main limitation of integrating ChatGPT with low-code platforms will be the potential difficulty of establishing a direct integration between the two parties. This combined solution should be able to implement a "use case" orientated approach in order to ensure transversal success. Hence, despite the complexity associated with knowing how to "ask the right question" or one that all parties understand well, this is critical to the referred "use case" orientation and, consequently, might be considered a decisive success factor. Thus, there is a direct relation between the developer's experience and obtained a response, as a more experienced developer should effectively know how to pose better questions. Bottom-line, the final decision should still be human-based. | Future directions. |
| Regardless, we must always consider individuals' resistance to change at all levels, from end users to developers. The organization's resistance to change is also a potential bias towards accepting and using the ChatGPT/Low-Code-based approaches. | Future directions. |
| Despite security issues, expect a vast (double-digit) increase in productivity. Fewer and fewer developers will be needed, but those required will have to be better and better. They will have to effectively and efficiently interpret the established requirements and communicate them, in a structured manner, to the development through instructions given to ChatGPT. Therefore, we can expect a decrease in the size of development teams and an increase in their skills, experience and specialization. | It needs further research to confirm. |

(continued on next page)

**Table 5** (*continued*)

| Topic | Remark |
|---|---|
| From the developer's perspective, the features they would like to see implemented are those related to tuning, detection of performance and load issues, detection of security problems, and documentation automation. In what concerns development accelerators, the referred individuals would like to see the ability to generate forms, generate screens, and promote the connectivity between elements and objects. Also, when addressing the test and debug features, developers would like to have access to fully automated testing with artificial intelligence (AI) engines generating and running all the necessary tests.<br>The introduction of AI in all the development process (from the coding to the testing stages) can also ensure the application of international norms and standards (usability, security, amongst others). | Future directions |
| Considering the context inherent to ChatGPT, low-code platforms, and the combination of these two artefacts, future training necessities should focus on prompt engineering instead of coding languages and techniques. | Future directions |

**Table 6**
Model improvements discussed.

| Improvement | Contribution |
|---|---|
| The focus group highlighted that ChatGPT could assist in the data modelling phase. Using the requirements collected by the human developer, ChatGPT can identify main concepts and their attributes, generating a draft model that would then be manually validated. | Inclusion of ChatGPT in the Data Modelling phase. |

usability and accessibility, performance, code uniformization, etc.. Furthermore, and drawing their opinion on authors such as Jalil et al. (2023), the focus group also perceived that ChatGPT could not only be used to train professionals on how to test and document software solutions, but also to perform those same tests, to automate them and to implement on-time bug fixing actions.

As argued by existing literature (Gozalo-Brizuela & Garrido-Merchan, 2023), ChatGPT has a very well-established set of features that allow it to perform code analysis and interpretation. With this in mind, the focus-group experts argue that with the aid of ChatGPT the consumption or exposure of APIs will become not only easier but also less time-consuming as the effort associated with both understanding the manner in which the APIs ought to be used and the effort, and the use itself of the APIs are decreased.

According to both the focus group experts and the analysed literature (Li et al., 2023), ChatGPT ability to ensure completeness in its responses still lacks a significant improvement. In what concerns the software development processes, and particularly the low-code/no-code developments, it was possible to collect consensus from the abovementioned experts that in order to fully address this issue it might be very relevant to incorporate ChatGPT throughout the entire development process, i.e. from both the initial phases (requirements analysis and modelling) to the last phases (testing, bug fixing and deployment).

According to authors such as Mahadi Hassan et al. (2023), the extent to which ChatGPT can deliver cohesive responses is yet to be further analysed as there is a significant rate of inconsistency and lack of precision in the content that the referred tool generates. During the focus group execution this topic has also been discussed, to the point of the

**Table 7**
Attributes for qualitative assessment.

| Attribute | Evaluation | Main contributions and perspectives |
|---|---|---|
| Clarity | 100 % | Although it requires a previous understanding of ChatGPT and what you can do with it, it is easy to understand the proposed model. |
| Complexity | 100 % | The complexity of the proposed model creates the necessity for a more practical guideline for operationalizing it. |
| Feasibility | 100 % | There are some concerns regarding who can efficiently implement the model. |
| Maintainability | 100 % | The model should be open to any platform and updated with contributions from any developer. |
| Completeness | 60 % | The model should provide for the use of ChatGPT at all stages, even in the user interview. |
| Consistency | 100 % | It is transversal to the development process and platform-neutral. |
| Cohesion | 60 % | Some aspects need to be clarified regarding what ChatGPT can contribute to each stage of the development process. |
| Scalability | 100 % | It is limited only by what ChatGPT, either in the current version or in future versions, can do efficiently. |
| Adaptability | 100 % | The use of ChatGPt can be done in a modular way and at the developer's decision at each stage. It can be applied to the development process on any low-code platform. |
| Prescriptive | 40 % | The model needs more detail, specifically on what ChatGPT can contribute at each stage of the development process. |
| Flexibility | 100 % | The use of ChatGPT at each stage will depend on the specific situation and should not be defined *a priori*. |
| Management Support | 100 % | There is immense potential for governance support in the development process and later maintenance stages. |

involved experts consensually established that, at this point in time and with ChatGPT current feature catalogue, it is significantly difficult to not only enforce the existing of cohesion in the content generation process, but also to assess the precision dimension at each phase of the development process. Hence, this is a topic that should clearly be address in a more focused manner in the near-future.

Drawing on the abovementioned perceptions and suggestions, a revised version of the initially proposed model was established (Fig. 7).

Hence, ChatGPT is a technology with the potential to open up new avenues in software development. From a global perspective, the focus group experts recognized the proposed model's interest and potential, despite their own lack of perception on the impact this tool might have on the employment market as it could possibly replace the human developer. Yet, the true potential of ChatGPT use still needs to be determined, and we are still learning what can and cannot be done with the tool. Nevertheless, there is a clear common belief that this new paradigm is the "new future".

## 6. Final considerations

Low-code platforms are a trend in software solutions development, mainly due to the many advantages those platforms can bring to the development process. However, together with all those advantages, several problems have been identified, from which we can easily highlight the need for software engineers and developers for evaluation and monitoring-related tasks, and the unpleasant experience from the developer's perspective related to potential work constraints, limited freedom and creativity, inadequate documentation and overview, and poor and unsafe teamwork capabilities.

Based on the challenges inherent to the abovementioned context, a research initiative has been executed in order to fully assess the pros and cons inherent to use of ChatGPT as a tool that would not only aid developers but also to minimize, if not solve, the typical problems

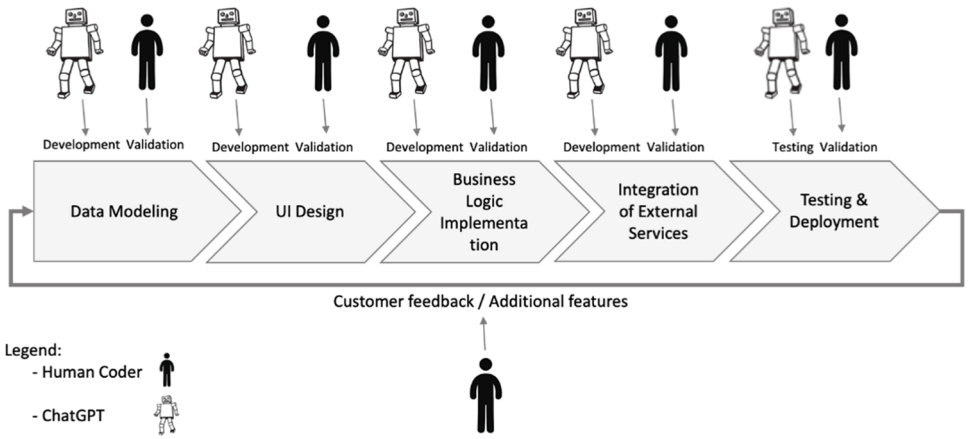



**Fig. 7.** Final version of the proposed model.

associated with software development. In order to deliver a proper contribute to the posed challenge, we propose a novel model, specially focused on low-code platforms, that establishes an active collaboration approach to the software development process that combines both humans and ChatGPT. To establish the potential validity and overall value of the proposed artefact, a focus group, involving experts from both the academia and the business contexts, has been performed and we were able to not only reach a consensual appraisal of the model but also a set of new perspectives and suggestions that were used to improve out initial proposal.

At this point of the work, we can list as main practical contributes of the proposed model, the following: the ability to establish a proper methodological approach to foster accelerated software development with reduced technical barriers where improved collaboration, efficient debugging and enhanced documentation are considered as positive drivers of success. Furthermore, the proposed model also ensures both software engineers and programmers skill augmentation and trigger their ability to adapt to novel challenges. The proposed artefact also encompasses the reduction in repetitive tasks and an improved cost-efficiency in the process of software development with low-code platforms.

From a theoretical scope our research poses as a novel set of knowledge, properly supported by both existing scientific and grey literature that other researchers can use as foundation for their own research on the use of artificial intelligence tools, such as ChatGPT, as technical and functional aid to low-code software development initiatives.

Hence, from a global perspective, this research significantly contributes to the future adoption of LCSD with the support of ChatGPT, thus triggering and accelerating the production of software solutions with higher levels of quality and standardization, by proposing a novel model that merges the efforts from both the developers and ChatGPT and that, despite being independent of the low-code platform itself, ensures that the training and expertise of the developer in using ChatGPT have a decisive role for the final outcome of the development process.

### *6.1. Limitations and future research*

After a careful analysis to the performed research we were able to identify the existence of some limitations that despite worthy of description, do not impail the overall value of the research nor of the achieved results.

Even thou the focus group approach is well established within the existing literature, there is also a relevant amount of research on the potential issues or limitations regarding this type of approach. Hence, with this in mind, we do believe that the decision to execute a focus group could have limited the density of the achieved results. In order to address this potential impairment, we propose that future research would encompasses the execution of a case study where developers and software engineers could combine ChatGPT with LCSD plataforms at the same time that their entire experience is closely monitored and controlled.

Furthermore, and considering the feedback from the focus group experts (highlighted in Tables 4 and 5, and detailed in Section 5), we believe that the future research should focus on the described topics, thus ensuring further enhancing and developing to the proposed model, and bringing it closer to a prescription model to be used by developers.

## 7. Conclusions

ChatGPT and LCSD are trends by themselves, and even more when we bring them to work together. With the proposed (and validated) model we establish a novel, innovated and structured approach to use a tool like ChatGPT, thus aiming at the increase in the effectiveness and efficiency already boosted by developing solutions based on low-code platforms.

## Funding

This work is financed by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project LA/P/0063/2020.

## Institutional review board statement

Not applicable.

## Informed consent statement

Not applicable.

## CRediT authorship contribution statement

**José Martins:** Conceptualization, Methodology, Investigation, Writing – review & editing, Supervision, Funding acquisition. **Frederico Branco:** Conceptualization, Methodology, Investigation, Writing – review & editing, Supervision, Funding acquisition. **Henrique Mamede:** Conceptualization, Methodology, Investigation, Writing – review & editing, Supervision, Funding acquisition.

## Declaration of Competing Interest

The authors declare the following financial interests/personal

relationships which may be considered as potential competing interests:

Jose Martins reports financial support was provided by Foundation for Science and Technology.

## Data availability

Data will be made available on request.

## Acknowledgements

We acknowledge the effort devoted to this work by all the experts participating in the Focus Group that served as a validation instrument for the proposed conceptual artefacts.

José Martins is grateful to the FCT for the projects titled "Aqua-Valor—Centro de Valorização e Transferência de Tecnologia da Água" (NORTE-01-0246-FEDER-000053) and "Emprego altamente qualificado nas empresas ou em COLABS—Contratação de Recursos Humanos Altamente Qualificados (PME ou CoLAB)" (NORTE-06-3559-FSE-000095), supported by the Norte Portugal Regional Operational Programme (NORTE 2020) under the PORTUGAL 2020 Partnership Agreement through the European Regional Development Fund (ERDF)

## References

Ahmad, A., Waseem, M., Liang, P., Fahmideh, M., Aktar, M.S., & Mikkonen, T. (2023). *Towards human-bot collaborative software architecting with chatgpt*. 279–285.

Al Alamin, M. A., Malakar, S., Uddin, G., Afroz, S., Haider, T. B., & Iqbal, A. (2021). An empirical study of developer discussions on low-code software development challenges. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)* (pp. 46–57). https://doi.org/10.1109/MSR52588.2021.00018

Aljanabi, M. (2023). ChatGPT: Future directions and open possibilities. *Mesopotamian Journal of CyberSecurity, 2023*, 16–17.

Beranic, T., Rek, P., & Hericko, M. (2020). Adoption and usability of low-Code/No-code development tools. Varazdin: Faculty of Organization and Informatics Varazdin. In *Proceedings of the Central European Conference on Information and Intelligent Systems* (pp. 97–103).

Beres, J. (2022). Can low-code tools end the developer shortage? *BuiltIn*. https://builtin.com/software-engineering-perspectives/low-code-tools-developer-shortage.

Bidochko, A. (2023, September 3). The power of ChatGPT in AI and low-code/no-code development: Driving great business outcomes [UBOS - Unified Business Operating System]. *UBOS - Blog*. https://ubos.tech/the-power-of-chatgpt-in-ai-and-low-codeno-code-development-driving-great-business-outcomes/.

Blanchard, C. (2013). Make or buy?: The software developer shortage that isn't. *The Blue Review*.

Bock, A. C., & Frank, U. (2021). Low-code platform. *Business & Information Systems Engineering, 63*(6), 733–740. https://doi.org/10.1007/s12599-021-00726-8

Bogdanova, D., & Snoeck, M. (2019). CaMeLOT: An educational framework for conceptual data modelling. *Information and Software Technology, 110*, 92–107. https://doi.org/10.1016/j.infsof.2019.02.006

Breaux, T., & Moritz, J. (2021). The 2021 software developer shortage is coming. *Communications of the ACM, 64*(7), 39–41.

Brockman, G., Eleti, A., Goerges, E., Jang, J., Kilpatrick, L., Lim, R., Miller, L., & Pokrass, M. (2023). *Introducing ChatGPT and whispers APIs*. OpenAI Blog. https://openai.com/blog/introducing-chatgpt-and-whisper-apis.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901.

Bucaioni, A., Cicchetti, A., & Ciccozzi, F. (2022). Modelling in low-code development: A multi-vocal systematic review. *Software and Systems Modeling, 21*(5), 1959–1981.

Burger, B., Kanbach, D. K., Kraus, S., Breier, M., & Corvello, V. (2023). On the use of AI-based tools like ChatGPT to support management research. *European Journal of Innovation Management, 26*(7), 233–241. https://doi.org/10.1108/EJIM-02-2023-0156

Cai, Y., Mao, S., Wu, W., Wang, Z., Liang, Y., Ge, T., Wu, C., You, W., Song, T., & Xia, Y. (2023). Low-code LLM: Visual Programming over LLMs. *ArXiv Preprint ArXiv: 2304.08103*.

Camarinha-Matos, L. M., Fornasiero, R., Ramezani, J., & Ferrada, F. (2019). Collaborative networks: A pillar of digital transformation. *Applied Sciences, 9*(24), 5431.

Carlbring, P., Hadjistavropoulos, H., Kleiboer, A., & Andersson, G. (2023). A new era in Internet interventions: The advent of Chat-GPT and AI-assisted therapist guidance. *Internet Interventions, 32*, Article 100621. https://doi.org/10.1016/j.invent.2023.100621

Casadei, R., Fortino, G., Pianini, D., Russo, W., Savaglio, C., & Viroli, M. (2019). Modelling and simulation of opportunistic IoT services with aggregate computing. *Future Generation Computer Systems, 91*, 252–262.

Chen, W.-E., Lin, Y.-B., Yen, T.-H., Peng, S.-R., & Lin, Y.-W. (2022). DeviceTalk: A no-code low-code IoT device code generation. *Sensors, 22*(13), 4942. https://doi.org/10.3390/s22134942

Combemale, B., Gray, J., & Rumpe, B. (2023). ChatGPT in software modeling. *Software and Systems Modeling, 22*(3), 777–779. https://doi.org/10.1007/s10270-023-01106-4

Conchúir, E.Ó., Ågerfalk, P. J., Olsson, H. H., & Fitzgerald, B. (2009). Global software development: Where are the benefits? *Communications of the ACM, 52*(8), 127–131. https://doi.org/10.1145/1536616.1536648

Dwivedi, Y. K., Kshetri, N., Hughes, L., Slade, E. L., Jeyaraj, A., Kar, A. K., Baabdullah, A. M., Koohang, A., Raghavan, V., Ahuja, M., Albanna, H., Albashrawi, M. A., Al-Busaidi, A. S., Balakrishnan, J., Barlette, Y., Basu, S., Bose, I., Brooks, L., Buhalis, D., … Wright, R. (2023). So what if ChatGPT wrote it?" Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy. *International Journal of Information Management, 71*, Article 102642. https://doi.org/10.1016/j.ijinfomgt.2023.102642

Elshan, E., Ebel, P., Söllner, M., & Leimeister, J. M. (2023). Leveraging low code development of smart personal assistants: An integrated design approach with the SPADE method. *Journal of Management Information Systems, 40*(1), 96–129.

Esposito, A., Calvano, M., Curci, A., Desolda, G., Lanzilotti, R., Lorusso, C., & Piccinno, A. (2023). *End-User Development for Artificial Intelligence: A Systematic Literature Review*, 19–34.

Gao, D. (2022). *Measuring developers' episodic experience of low-code development platforms* [Master Degree Thesis, Aalto University] https://aaltodoc.aalto.fi/handle/123456789/117387.

Gottschalk, S., Bhat, R., Weidmann, N., Kirchhoff, J., & Engels, G. (2022). *Low-code experimentation on software products*. 798–807.

Gozalo-Brizuela, R., & Garrido-Merchan, E.C. (2023). ChatGPT is not all you need. A State of the Art Review of large Generative AI models. *ArXiv Preprint ArXiv:*2301.04655.

Gonçalves, R. M., Martins, J., Branco, F., Cota, M. P., & Oliveira, M. A. (2016). Increasing the reach of enterprises through electronic commerce: A focus group study aimed at the cases of Portugal and Spain. *Computer Science and Information Systems, 13*, 927–955.

Guest, G., Namey, E., & McKenna, K. (2017). How many focus groups are enough? Building an evidence base for nonprobability sample sizes. *Field Methods, 29*(1), 3–22. https://doi.org/10.1177/1525822x16639015

Guo, B., Zhang, X., Wang, Z., Jiang, M., Nie, J., Ding, Y. Yue, J., & Wu, Y (2023). *How close is ChatGPT to human experts? Comparison corpus, evaluation, and detection*. https://doi.org/10.48550/ARXIV.2301.07597.

Gursoy, D., Li, Y., & Song, H. (2023). ChatGPT and the hospitality and tourism industry: An overview of current trends and future research directions. *Journal of Hospitality Marketing & Management*, 1–14.

Hennink, M. M. (2014). 35Designing and conducting focus group research. M. M. Hennink & P. Leavy. *Understanding focus group discussions*. Oxford University Press. https://doi.org/10.1093/acprof:osobl/9780199856169.003.0002.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2008). Design science in information systems research. *Management Information Systems Quarterly, 28*(1), 6.

Jafarinejad, F. (2023). Synset2Node: A new synset embedding based upon graph embeddings. *Intelligent Systems with Applications, 17*, Article 200159. https://doi.org/10.1016/j.iswa.2022.200159

Jalil, S., Rafi, S., LaToza, T. D., Moran, K., & Lam, W. (2023). ChatGPT and Software Testing Education: Promises & Perils. In *2023 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. https://doi.org/10.48550/ARXIV.2302.03287

Johnsson, B. A., & Magnusson, B. (2020). Towards end-user development of graphical user interfaces for internet of things. *Future Generation Computer Systems, 107*, 670–680. https://doi.org/10.1016/j.future.2017.09.068

Juhas, G., Molnar, L., Juhasova, A., Ondrisova, M., Mladoniczky, M., & Kovacik, T. (2022). Low-code platforms and languages: The future of software development. In *2022 20th International Conference on Emerging ELearning Technologies and Applications (ICETA)* (pp. 286–293). https://doi.org/10.1109/ICETA57911.2022.9974697

Kidd, P. S., & Parshall, M. B. (2000). Getting the focus and the group: Enhancing analytical rigor in focus group research. *Qualitative Health Research, 10*(3), 293–308.

Köbis, N., & Mossink, L. D. (2021). Artificial intelligence versus Maya Angelou: Experimental evidence that people cannot differentiate AI-generated from human-written poetry. *Computers in Human Behavior, 114*, Article 106553. https://doi.org/10.1016/j.chb.2020.106553

Krishnaraj, N., Vidhya, R., Shankar, R., & Shruthi, N. (2022). *Comparative study on various low code business process management platforms* (pp. 591–596). IEEE.

Krueger, R. A., & Casey, M. A. (2002). *Designing and conducting focus group interviews, 18*. The World Bank.

Li, H., Wang, W., Liu, Z., Niu, Y., Wang, H., Zhao, S., Liao, Y., Yang, W., & Liu, X. (2022). A novel locality-sensitive hashing relational graph matching network for semantic textual similarity measurement. *Expert Systems with Applications, 207*, Article 117832. https://doi.org/10.1016/j.eswa.2022.117832

Li, X., Jiang, Y., & Mostafavi, A. (2023). AI-assisted protective action: Study of ChatGPT as an information source for a population facing climate hazards. *ArXiv Preprint ArXiv:*2304.06124.

Liu, C., Bao, X., Zhang, H., Zhang, N., Hu, H., Zhang, X., & Yan, M. (2023). Improving ChatGPT prompt for code generation. *Software Engineering*. https://doi.org/10.48550/arXiv.2305.08360arXiv:https://doi.org/10.48550/arXiv.2305.08360

Luo, Y., Liang, P., Wang, C., Shahin, M., & Zhan, J. (2021). Characteristics and challenges of low-code development: The practitioners' perspective. *Software Engineering*, 1–11.

Mahadi Hassan, M., Knipper, A., & Kanti Karmaker Santu, S. (2023). ChatGPT as your Personal Data Scientist. *ArXiv E-Prints*. arXiv-2305.

Mather, C., & Cummings, E. (2017). Modelling digital knowledge transfer: Nurse supervisors transforming learning at point of care to advance nursing practice. *Informatics, 4*(2), 12.

Mohamad Suhaili, S., Salim, N., & Jambli, M. N. (2021). Service chatbots: A systematic review. *Expert Systems with Applications, 184*, Article 115461. https://doi.org/10.1016/j.eswa.2021.115461

Morgan, D. L. (1996). Focus groups. *Annual Review of Sociology, 22*(1), 129–152.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback. *Computation and Language*. https://doi.org/10.48550/ARXIV.2203.02155

Pan, S. L., & Nishant, R. (2023). Artificial intelligence for digital sustainability: An insight into domain-specific research and future directions. *International Journal of Information Management, 72*, Article 102668. https://doi.org/10.1016/j.ijinfomgt.2023.102668

Philips, T. (2022). Is there a shortage of developers? Developer shortage statistics in 2022. *Code Submit. Industry Research Report* https://codesubmit.io/blog/shortage-of-developers/.

Richardson, C., Rymer, J. R., Mines, C., Cullen, A., & Whittaker, D. (2014). *New development platforms emerge for customer-facing applications* (p. 15). Cambridge, MA, USA: Forrester.

Rizou, S., Theofilatos, A., Paflioti, A., Pissari, E., Varlamis, I., Sarigiannidis, G., & Chatzisavvas, K. Ch (2023). Efficient intent classification and entity recognition for university administrative services employing deep learning models. *Intelligent Systems with Applications, 19*, Article 200247. https://doi.org/10.1016/j.iswa.2023.200247

Sanchis, R., García-Perales, Ó., Fraile, F., & Poler, R. (2019). Low-code as enabler of digital transformation in manufacturing industry. *Applied Sciences, 10*(1), 12.

Santos, O. C., & Boticario, J. G. (2015). User-centred design and educational data mining support during the recommendations elicitation process in social online learning environments. *Expert Systems, 32*(2), 293–311.

Sarkar, T. (2022). XBNet: An extremely boosted neural network. *Intelligent Systems with Applications, 15*, Article 200097. https://doi.org/10.1016/j.iswa.2022.200097

Sloyan, T. (2021). Is there a developer shortage? Yes, but the problem is more complicated than it looks. *Forbes Technology Council. June, 8* https://www.forbes.com/sites/forbestechcouncil/2021/06/08/is-there-a-developer-shortage-yes-but-the-problem-is-more-complicated-than-it-looks/?sh=4ece29193b8e.

Sobania, D., Briesch, M., Hanna, C., & Petke, J. (2023). *An analysis of the automatic bug fixing performance of ChatGPT*. Cornell University ArXiv, Software Engineering. https://doi.org/10.48550/ARXIV.2301.08653

Stewart, D.W., & Shamdasani, P.N. (1998). *Focus group research: Exploration and discovery*.

Sundberg, L., & Holmström, J. (2023). Democratizing artificial intelligence: How no-code AI can leverage machine learning operations. *Business Horizons*. https://doi.org/10.1016/j.bushor.2023.04.003

Tian, H., Lu, W., Li, T.O., Tang, X., Cheung, S.-C., Klein, J., & Bissyandé, T.F. (2023). Is ChatGPT the ultimate programming assistant—How far is it? *ArXiv Preprint ArXiv:*2304.11938.

Tiwari, A., Khandwe, A., Saha, S., Ramnani, R., Maitra, A., & Sengupta, S. (2023). Towards personalized persuasive dialogue generation for adversarial task oriented dialogue setting. *Expert Systems with Applications, 213*, Article 118775. https://doi.org/10.1016/j.eswa.2022.118775

Wang, F.-Y., Miao, Q., Li, X., Wang, X., & Lin, Y. (2023). What does ChatGPT say: The DAO from algorithmic intelligence to linguistic intelligence. *IEEE/CAA Journal of Automatica Sinica, 10*(3), 575–579. https://doi.org/10.1109/JAS.2023.123486

Wang, S., Scells, H., Koopman, B., & Zuccon, G. (2022). Automated MeSH term suggestion for effective query formulation in systematic reviews literature search. *Intelligent Systems with Applications, 16*, Article 200141. https://doi.org/10.1016/j.iswa.2022.200141

Xu, L., Sanders, L., Li, K., & Chow, J. C. (2021). Chatbot for health care and oncology applications using artificial intelligence and machine learning: Systematic review. *JMIR Cancer, 7*(4), e27850.