IMPACT OF ADOPTING AI TOOLS BY SOFTWARE DEVELOPERS TOWARDS PRODUCTIVITY AND SUSTAINABILITY

Lappeenranta–Lahti University of Technology LUT

Master’s Programme in Software Product Management & Business and Master's thesis

2024

Mohammad Asif Hassan

Examiner(s): Professor Jari Porras

Post-Doctoral Resracher Sanaul Haque

ABSTRACT

Lappeenranta–Lahti University of Technology LUT

LUT School of Engineering Science

Software Engineering

Mohammad Asif Hassan

IMPACT OF ADOPTING AI TOOLS BY SOFTWARE DEVELOPERS TOWARDS PRODUCTIVITY AND SUSTAINABILITY

Master's thesis

2024

73 pages, 8 figures, 5 tables and 2 appendices

Examiner(s): Professor Jari Porras and Post-Doctoral Researcher Sanaul Haque

Keywords: AI Tools, Artificial Intelligence, Software Developer

Software developers are adopting artificial intelligence (AI) tools to develop software products and promote their software business. Every software project has three requirements: they must be finished on schedule, within budget, and to the highest standard of quality achievable. In the software development process, Software developers face manual and time-consuming coding tasks, Inadequate bug detection and resolution, complexity in code refactoring, and limited test coverage and maintenance. Recently, software developers have harnessed the power of AI tools to create innovative solutions to achieve their desired milestones and mitigate software development complexities in their working premises. This research uses a mix-method approach, including a systematic literature review, and a few semi-structured interviews to find out the key factors, benefits, and barriers of adopting AI tools and the deep thoughts of the software developers who worked in a scrum team and their contributions towards sustainability in the software development process respectively. This study found that how AI tools help developers to enhance code quality, reduce cognitive load and boost the testing through auto code

suggestions, auto task completion, code refactoring, innovative suggestions and automate testing process and assist developers in contributing to sustainability by reducing development time and cost. This research provides a solid idea, of how software developers are using AI tools in their working environments to mitigate their workloads to develop the best quality software products after having barriers and their contributions towards sustainability.

ACKNOWLEDGEMENTS

I am very thankful to LUT University to provide an opportunity to work on this thesis with the most intellectual people around me. This master's program provides me the good time to complete my thesis under the supervision of Jari and Sanaul. They always inspire me to get on the right track and provide invaluable guidance throughout the whole path. Their ingenious opinion has made a contribution shaping my thesis and enhancing the quality of my work.

I am always grateful to my family members, teachers, and friends who have provided support in my hard time on this journey to achieve my goal and personal growth. This achievement is not possible without all of you.

Mohammad Asif Hassan

LUT University, Finland

06 December 2024

DECLARATION OF AI USAGE

Grammarly: Proofread my content, Correction of Grammar, clarity and readability, Punctuation

ChatGPT: Clarifying concepts, Understanding Context

QuillBot: To check grammar, paraphrasing, and improve writing

I declare that no other AI assistance or tools were used in the development of the contents of my thesis work than the above-mentioned.

ABBREVIATIONS

Artificial Intelligence (AI)

Software Engineering (SE)

Systematic Literature Review (SLR)

Deep Learning (DL)

Retrieval-Augmented Generation (RAG)

Machine Learning (ML)

Software Development Life Cycle (SDLC)

Natural language Processing (NLP)

Automated Program Repair (APR)

Quality Assurance (QA)

Large Language Model (LLM)

Generative Pre-Trained Transformer 2(GPT-2)

Software Project Management (SPM)

Table of Contents

Abstracts

Acknowledgements

Declaration of AI Use

Abbreviations

1 INTRODUCTION 1

2 BACKGROUND AND RELATED WORKS 4

2.1 AI Tools in Software Development 4

2.2 AI Tools Scrum Framework in Software Development Process 5

2.3 Impact of AI and Scrum in Sustainability 6

2.4 AI Tools in the Advancement of Software Systems 7

2.5 Key Factors, Benefits, and Barriers of Adopting AI Tools 11

2.5.1 Key Factors of Adopting AI Tools in Software Development 12

2.5.2 Benefits of Adopting AI Tools 15

2.5.3 Barriers to Adopting AI Tools 19

2.6 AI Tools in Scrum and Sustainable Software Development 22

2.7 Research Gap & Problem Statement 25

3 METHODOLOGY 26

3.1 Research Method 26

3.2 Systematic Literature Review 28

3.2.1 Search Strategy 29

3.2.2 Selection Criteria 29

3.2.3 Data Extraction 32

3.2.4 Data Analysis 32

3.3 Interview Design 33

3.3.1 Recruitment Process & Participants 33

3.3.2 Interview Format 36

3.3.3 Interview Questions & Guideline 36

3.3.4 Data Collection & Recording Process 36

3.3.5 Interview Analysis 37
4 RESULTS 37
4.1 Research Question 1 (SLR) 38
4.1.1 Key Findings From SLR 38
4.2 Research Question 2 (Interview-based) 42
4.2.1 Key Findings from Interview 42
5 DISCUSSION 46
5.1 Restating Research Aim and RQs 47
5.2 Key Findings 48
5.2.1 Comparing with Existing Literature and Real-life Practice 49
5.2.2 Explain Unexpected Findings 51
5.2.3 Discussion of Implication of Findings 51
5.3 Limitations of the Study 52
5.4 Future Works 53
6 CONCLUSIONS 53
References 56

APPENDICES

A APPENDIX: SOURCE OF SLR KEY FINDINGS

B APPENDIX: INTERVIEW QUESTIONS

FIGURES

Figure 1: Scrum Process Flow

Figure 2: AI Tools in Software Development Life Cycle

Figure 3: Highlighting Different AI Tools Adopted as a Key Tool

Figure 4: The Research Process Diagram

Figure 5: Literature Review Process

Figure 6: Identification of Benefits and Barriers of Adopting AI Tools from SLR

Figure 7: Identification of Advantages of AI tools in software Developers' workflow and contribution to Sustainability

Figure 8: Impact of AI Tools on Productivity and Sustainability

TABLES

Table 1: Comparison between AI Coding Assistant Tools

Table 2: Impact of AI Tools on the Scrum Process

Table 3: Inclusion and Exclusion Criteria for Selecting Studies

Table 4: Recruitment Process for Selecting Participants

Table 5: List of participants selected for Interview

1 INTRODUCTION

Over the last few decades, artificial intelligence (AI) has evolved and transformed software engineering in accuracy, productivity, and collaboration by offering new AI tools and techniques in software development practices (Waseem et al., 2024). Nowadays, AI coding assistant tools such as ChatGPT, GitHub Copilot, Tabnine, Stack Spot AI, Rakuten AI etc are used in software development that provide code suggestions to software developers to reduce mistakes and human efforts (N. Davila et al., 2024). Traditional software development tools are labour-intensive and consume more time to develop software products. However, to accomplish a software project, its goals are to deliver on time, within an estimated budget and acquire the highest standard of quality in software. After addressing the pain points of manual efforts, Software Developers are adopting AI tools to avoid repetitive tasks, and manual workload and provide innovative solutions. However, there is limited research about the factors that influence software developers to adopt AI tools like GitHub Copilot, ChatGPT, Tabnine, Stack Spot AI, and Rakuten AI etc, though these tools have limitations. Furthermore, there are unexplored areas that show how software developers are using AI tools to contribute to sustainability in software development processes like Scrum.

The software developers use AI tools to optimize their coding tasks, refactor code to boost sprint activity in the scrum process. There is limited research that focuses on security vulnerabilities of AI tools (ChatGPT, Copilot) generated codes and few types of research indicate the benefits of adopting the AI tools (Klemmer & Jan, 2024) in the software development process. However, these research don’t discuss the influential matters, key points, benefits, and barriers of adopting AI tools by software developers. Besides, these research papers don’t discuss how these AI tools increase the productivity of software developers by providing code suggestions, code reviews, code refactoring, and automated testing. There is no clear investigation that AI tools have a broader impact in the scrum process to boost the software development process, and developers are using AI-generated code by following sustainable coding practices to reduce environmental impact, development time, and cost to bring sustainability, respectively.

This study elaborately presents how AI tools create impact on software developers to enhance their skills, avoid mundane tasks, and increase their productivity in code optimization, code refactoring, and bug fixing in software projects. This research discovers the scenarios that limit the developers from adopting AI tools and investigates the insights of the software developers on how these AI tools promote sustainability by reducing development time and cost in a scrum.

This research aims to investigate the key factors, benefits, and barriers associated with AI tools adopted by software developers in the process of creating software. It reveals the perspectives of the software developers on how these AI tools boost productivity by automating tasks, optimizing code, and contributing to sustainability by reducing development time and cost in the scrum. This research formulates two research questions to achieve the goal.

With a view to tracing out the inclination of software developers to engage in using AI tools, in order to develop codes, code refactoring, and testing, a Systematic Literature Review (SLR) has been earnestly applied. This study investigates the limitations of AI tools, which are prominent constraints for software developers to use in the development phase of software. This research also applies a semi-structured interview approach to understand the actual experiences of the software developers to adopt the Artificial Intelligence tools in their software creation cycle, how these Artificial Intelligence tools increase their productivity compared to traditional tools, and discover the fact that attribute is sustainability by reducing software development time and cost in the Scrum.

The main Contributions of this research are as follows:

The deeper insights that motivate software developers to adopt AI tools in the software development process.

Investigating how AI tools enhance productivity of the software developers.

Reveal The details of AI tools that aid software developers to contribute to sustainability.

The rest of the section of this study is structured in the following manner:

Chapter 2: The background contains a description of the most important aspects of this research, as well as a framing of the context and Related Works reviewing the existing literature to explore the key factors, benefits and barriers of using AI tools. This chapter also investigates how AI tools increase the productivity of software developers and contribute to sustainability and research gaps.

Chapter 3: Presents the methodology of the research: systematic literature review, interview design, data collection method and data analysis techniques.

Chapter 4: Presents the result generated from SLR based on RQ1 and the semi-structured interview process based on RQ2.

Chapter 5: Presents the Discussion of the key findings, Implication of Findings, Limitations of the study and future works.

Chapter 6: Conclusions - Presents the summary of key findings, contributions, limitations, and future work opportunities.

2 BACKGROUND AND RELATED WORKS

Artificial intelligence brings an immense revolution in its application to different industries and software companies, taking full advantage of AI technology to lead the greatest innovation. Nowadays, software developers are adopting a great number of AI tools for grasping and handling the complicated nature of software engineering (Gustavo et al., 2024). Integration of AI technologies in the Software Engineering (SE) domain has completely brought changes in software development procedures, especially in project management, modelling, development, software testing, and maintenance. (Nathalia et al., 2023). The AI tools are using their potential to optimize the software development process, automate tasks, reduce the development time and enhance the quality of the software. (Peras et al., 2023; Waseem et al., 2024). The advancement of AI technologies into software engineering is contributing to the software development process, notably in agile development approaches such as Scrum which is renowned for its adaptability, iterative procedures, and collaborative team dynamics.

2.1 AI Tools in Software Development

Conventional software development is manual, consumes a long time and requires extensive human efforts to complete the process. However, with the rise of AI techniques like automated code suggestion, code generation, code review, code refactoring, code snippets and automated testing, developers now have tools to accelerate the software development process, testing, issue fixing, and deployment. Nowadays, Generative AI (ChatGPT, Gemini), AI coding Assistant tools (Tabnine, Intellicode, Github Copilot, Stack SPot AI, DeepCode, Amazon CodeWhisperer, Sonar Cube, CURE) (Daniel at el., 2022) are the most frequently used Artificial Intelligence tools and these AI Assistants employ a large language model (LLM) to produce results for their users after being trained on hundreds of millions of lines of source code (Wendy et al., 2024). These tools offer real-time code suggestions, detect vulnerabilities, and automate testing tasks for software developers. AI tools are adopted in the software development process to increase the efficiency of developers and

reduce human errors. This tendency is especially advantageous in agile frameworks like Scrum, where AI's capacity to manage both regular and complex tasks may boost the quick, iterative nature of the job.

2.2 AI Tools Scrum Framework in Software Development Process

The Scrum framework, pioneered by Jeff Sutherland in the early 1990s, is an Agile software development methodology that improves focus, clarity, and transparency within project teams. It conforms to Agile principles, emphasizing customer happiness, cooperation, functional software, simplicity, and frequent team reflection. Scrum distinguishes itself from other Agile Methodologies such as XP and Lean (V. T. Faniran et al., 2017). In Scrum, it ensures collaboration between the team members through meetings, by assigning roles to manage and achieve the project goal. It is a platform to manage projects, focusing on recursive progression by means of events like sprint planning, scrutiny and demonstration. It makes legitimate use of product backlog and sprint backlog in order to ensure orderly management of relevant tasks and to achieve predefined goals via virtue of organized teamwork. (R. Mdallal, 2023).

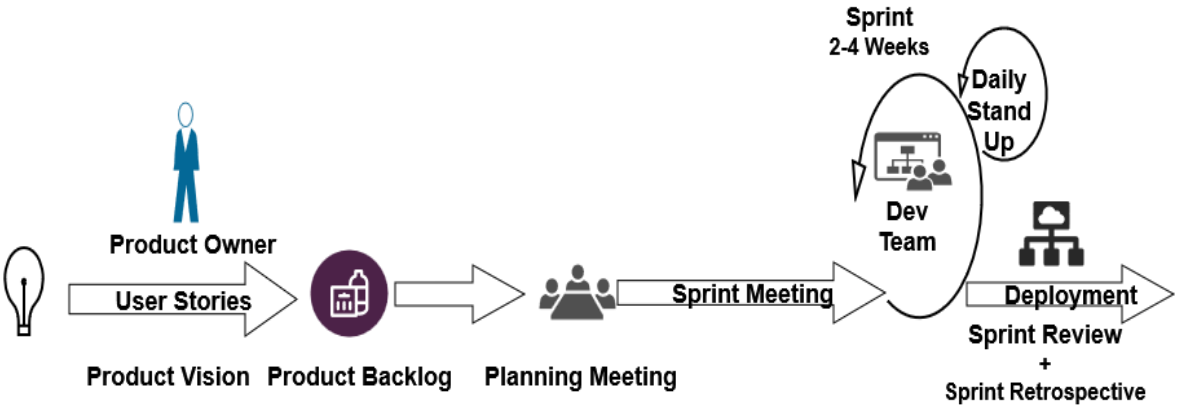

Figure 1: Scrum Process Flow (R. Mdallal, 2023)

Scrum is the most popular agile software development framework. Cross-functional scrum teams always focus on delivering products to customers. Scrum masters, product owners, and developers are the key roles in the scrum project, and they handle complex projects. While AI tools are integrated into scrum projects, they improve productivity, give real-time

data in sprint and automate repetitive tasks. To boost the sprint activities, different AI tools are used such as Stepsize AI, Github Insights, Jira with Jira Align etc. Though AI reduces the cognitive load of the developers however It needs further investigation that AI tools have an impact on the productivity of the software developers and boost their skills in sprints.

2.3 Impact of AI and Scrum in Sustainability

The new invention of technology has blessed our daily way of life and besides, the over-reliance on technologies raises risks in society such as the digital divide, isolation, cybersecurity and privacy. Therefore, the development of sustainable software is very crucial. A distinct feature of sustainability is to make sure required supply for ongoing needs, without depriving future generations' capability to ensure theirs. By mitigating the adverse effects during the software development, deployment and utilization in an environmentally sustainable way, sustainable software protects the interests of individuals, the economy, society and the environment. (B. Ayoola, 2023). Nowadays, sustainability is the most common and empirical topic of discussion in software and requirement engineering. In the software engineering domain, in Agile software development, scrum is the most popular and commonly used framework. It is clear that very few research are available that support the development of sustainable software systems by using agile methodologies, especially the scrum framework (P. Bambazek et al., 2023). Sustainability in software development is a new topic, but it has emerged as crucial. It is not only concerned about the environmental dimensions but also the technical, social and economic impact of software products. The goals of sustainable software development are to minimize the usage of resources, reduce e-waste and develop software that is scalable, maintainable and reduce the usage of energy.

In the scrum framework, the inclusion of Artificial Intelligence (AI) tools may reduce the production time of the software developers, stop repetitive tasks, automate tasks, reduce human dependency, reduce development costs, and lower the consumption of energy usage. Besides AI tools help to reduce excessive testing, refactoring, and rework indicating sustainable practices of SE. However, in social dimensions, AI tools reduce repetitive tasks, which helps software developers focus on other development-related activities and contribute to a more sustainable working environment.

2.4 AI Tools in the Advancement of Software Systems

Nowadays, Artificial intelligence (AI) is one of the popular and desirable subdivisions of computer science. AI helps the machine to act as an intelligent part of the system and enables them to work multiple, diverse tasks as their priority. Referring to (F. Sahin, 2022), AI acts as a catalyst that reduces human efforts to perform tasks and provides much betterment in an iterative manner based on the information they possess. Artificial intelligence is such a subject matter that is enthusiastically in discussion in both non-academic and academic periphery. AI Techniques have a dominant presence (K. Bhandari et al., 2023) and profoundly contribute to the software development life cycle (SDLC). The AI tools are able to have a strong footprint in the SDLC phases. The below schema depicts the impingement of AI in various phases of SDLC in Software Engineering.

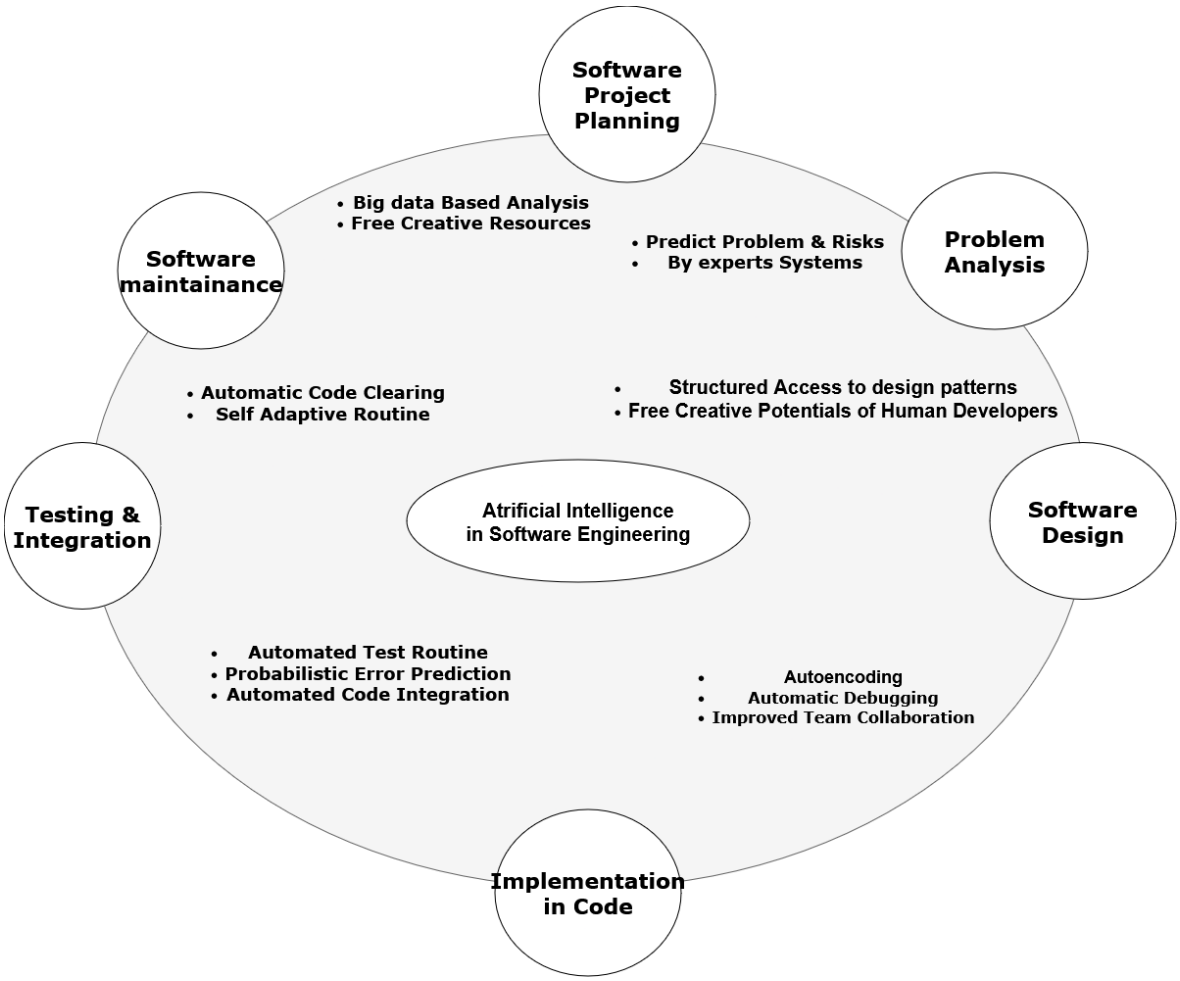

Figure 2: AI Tools in Software Development Life Cycle (Barenkamp et al., 2020)

The software developers are facilitated with various ranges of AI tools in their particular software development process from software design to maintenance to automate various phases in SDLC. ChatGPT is one of the most popular generative AI and fastest-growing users. It was developed by openAI and it had 100 million users in January 2023 and it has regular visitors 25 million (Nathalia et al., 2023). ChatGPT has gained popularity among software developers because of its automation capabilities for software development tasks such as coding testing (D. Sobania et al., 2023). ChatGPT helps developers work more productively and complete their duties more quickly. It is a subject of inquiry that ChatGPT surpasses human programmers regarding the quality of code and results (S. Imai, 2022). ChatGPT has been schooled in a myriad of computer languages and can deal with vast amounts of data. ChatGPT generates code according to the requirements of the software developers so that developers can polish the code and increase the standard of the code (Tian et al., 2023). Though ChatGPT has limitations, and it is not possible to replace software development work 100% by ChatGPT it is a key tool for software developers to enhance the software development workflows (J. W. Purwoko et al., 2023) in code generation, code refactoring and code explanation.

GitHub Copilot is a ground-breaking AI-powered software development tool, jointly contributed by GitHub and present-time sensation OpenAI. It is trained in several programming languages so that it generates correspondent codes in different languages, and it helps developers to write code faster and of higher quality (N. Nguyen & S. Nadi, 2022). Github Copilot functions as an alternative for pair programming. In software development, two programmers collaborate to come up with a functional code execution. GitHub Copilot offers the same benefits as human pair programming, and integrating this approach into software development enhances value, boosts productivity, and decreases both development time and the supplementary expense of employing a second developer (S. Imai, 2022). It is more intricate to detect the performance of GitHub Copilot in the creation of software than the human counterpart, but it provides multiple language code, function & class completion, fastest creation of boilerplate code, automatic code suggestions, reduced syntax error, auto bug detections etc. its recolonizations the software development process. (Rudolf et al., 2024) Illustrates that GitHub Copilot X is the next-generation Copilot code completion tool.

It has advanced features e.g. chat-based interaction that communicates directly with the development environment. It can provide real-time code assistance, give suggestions for improvements and it can debug, identify potential problems and fix errors.

Stack Spot AI is another contextualized coding assistant tool developed by a Brazilian tech company named Zup Innovation. Its knowledge source-based AI tool that enriches the knowledge process and is technically called Retrieval-Augmented Generation (RAG) (Gustavo et al., 2024). RAG always access real-time updated information that is trustworthy, transparent and verifiable. RAG is equipped with a retrieval and generation ability, so software developers can access high-quality code suggestions, configuration and explanation in real time (Gustavo et al., 2024). Alibaba developed an AI coding assistant tool named Lingma on 10 October 2021 and that provides plugins to visual studio code, JetBrains IDEs. This tool is trained on GPT-2 and the Elasticsearch. It helps software developers to generate code, repair bugs, code commenting, and test case generation. It makes software developers more productive by letting them look for code right in the integrated development environment (IDE), so they don't have to switch browsers when they're having trouble with programming (Chao at el., 2024). CodeVista is an AI coding assistant tool pre-trained on OpenAI's GPT-3.5 and GPT-4, Meta's LLaMA 2, and Google's Code Chat Bison and Gemini. It assists developers with coding chores via a chat interface and pre-established workflows. CodeVista incorporates Google Search, enabling users to locate analogous code samples or acquire further information regarding the coding concepts discussed (Khanh et al., 2024). Tabnine is a code completion tool developed by Codota. It is trained on 3 different types of Machine Learning (ML) to complete their tasks. It supports 25 different programming languages. It's code completion with deep learning. Intellicode focuses on Contextual IntelliSense and suggestions. It mainly supports IDEs Visual Studio and Visual Studio code. It influences refactoring suggestions and improves code style. Kite is another AI tool that performs autocomplete and in-line documentation. It supports Python and JavaScript (Roland, 2021). Stepsize AI is a good AI tool for effortless sprint reporting. It can generate automated reports, project risks and overall project health. This tool helps to enhance productivity and decision-making and creates better collaboration among team members. Bugasura is a user-friendly and intelligent bug-tracking AI tool. It ensures to spend

less time on bug detection and duplicate reports. The below table 1 shows the comparison between prominent AI tools used in software development.

Table 1: Comparison between AI Coding Assistant Tools

Feature	**ChatGPT**	**GitHub Copilot**	**Stack Spot AI**	**Lingma**	**Tabnine**
Primary Focus	AI-powered conversatio-nal assistant	AI-powered code completion and suggestion	AI-powered code generation, templates, and automation	AI for automated workflows and code generation	AI-driven code autocompleti on
Integration	Accessible via web or API	Integrated with Visual Studio, VS Code, JetBrains IDEs	DevOps platforms, project managemen t systems	Visual Studio, VS Code	Available in VS Code, JetBrains IDEs, Sublime
Code Completion	Can generate code based on queries	Autocomplet e based on context	Offers smart code snippets and templates	Autocomple tes based on patterns	Suggests entire lines or blocks of code
Contextual Code Assistance	understand complex queries	Based on the surrounding code	With retrieval-augmented generation (RAG)	Understands project-specific patterns	using deep learning in various languages

Code Documentation	Can explain and generate documentation	Can provide comments and simple explanations	Offers in-context documentation suggestions	Provides inline comments and documentation	Supports inline documentation lookup
Code Refactoring	Can suggest improvements when queried	Offers simple refactoring suggestions	Can refactor based on best practices retrieved	Provides refactoring hints	Basic refactoring suggestions based on patterns
Strengths	Flexibility, multi-purpose, can explain code in-depth	Real-time code completion, integrated with GitHub	Code templates, DevOps integration, enterprise-level automation	Workflow automation, DevOps and CI/CD focus	Versatile autocompletion across languages

2.5 Key Factors, Benefits, and Barriers of Adopting AI Tools

Incorporation of AI tools in the development phase of software has remarkably enriched the software industry and software developers are at ease. At present time software developers are being assisted by various coding tools and these tools are devised to let the developers navigate and break the complexity of software development (Yan Chen et al., 2017). The software industry heading towards the automation process with a view to improving both efficiency and quality of software. The AI coding assistant tools may provide inaccurate or

generic responses that hamper the development processes, despite their groundbreaking nature (Gustavo et al., 2024).

2.5.1 Key Factors of Adopting AI Tools in Software Development

Nowadays, AI tools are working as a torque in the creation of software items. The software developers embrace the power of the AI tools in their working environment. There are several key factors that help developers choose AI tools in their workflows to generate codes, reduce errors and better-quality code.

AI Tools in Software Development Life Cycle:

Artificial intelligence (AI), perhaps a principal accelerator of transformation, is transferring cognitive autonomy from developers to pre-trained models for diverse applications. One of the researchers investigates a comprehensive literature review by following SWEBOK guidelines in SDLC phases and connecting phases through Machine Learning (ML) types, tools and techniques (S. Shafiq et al., 2021). Only a few research are seen in the SDLC phases where AI tools have an impact from requirement elicitation to maintenance. AI tools automatically can prioritize the features and push them to the top of the to-do list. At the design phase, a prototyping tool named Figma is deployed to generate mockups and UI roadmaps, and it incorporates AI plugins, to fabricate UI elements automatically thus bringing about an amazing scope to construct AI prototyping tools. When the developers are not able to prioritize, the BERT tool can help them to place them. The best feature of the AI tools (i.e. GitHub Copilot, Tabnine) in software development is to reuse the code and code can be used in later sections or entirely in other projects (E. Trinh et al., 2024). AI tools help developers to generate test cases that reduce test case development time, and cost (H. Hourani et al., 2019). AI tools are also handy in terms of tracing any anomaly or incongruence and tracing out the responsible cause of any system glitch. No research is available to state AI tools' significant footprint on the due deployment and regular maintenance phase in SDLC.

Contextualized AI Coding Assistance Tool:

ChatGPT as a large language model (LLM) tool significantly improves the software development processes. ChatGPT is in the challenge when it encounters content-specific or specific answers. Without proper information, LLM is not able to generate the expected response as the user needs. Stack Spot AI is an AI contextualized coding assistant tool that was developed to solve contextual complexities and the software development process such as inconsistent responses, and lack of domain-specific knowledge (Gustavo et al., 2024). Contextualized coding assistance tools don't provide generic code completion ideas, it provides code suggestions that are fit to the current module. Stack Spot AI can engage different programming platforms that provide clarifications and insights and increase the developer's skills through interactive guidance (Steven et al., 2023).

Boost Productivity and Quality:

AI tools simultaneously improve the efficiency and productivity of software developers as well as code quality in an appreciative manner in the development process throughout the SDLC by automating repetitive tasks, better code accuracy and assisting developers. These tools create an environment in the SDLC phase to detect errors in code, review code and test (Saklamaeva & Pavlič, 2024). (Gustavo et al., 2024), illustrates that Stack Spot AI overcomes the general purpose of software development tools and saves time by delivering speedy and contextual small blocks of code so that the developer can concentrate on more creative or complex tasks. In some cases, the Generative AI, ChatGPT outperforms humans in terms of the quality of the code in the development workflows (Nathalia et al., 2023). GitHub Copilot also automatically suggest code snippets to lead to faster coding and it also reduces development time and boosts productivity by boilerplate coding. As a Generative AI, ChatGPT improves the patterns of code and refactoring the code to enhance the quality of code. This generative AI is also good for auto-code generation and code documentation (Waseem et al., 2024).

Automation & Prioritization of Tasks:

The automation process is the key feature of AI tools. One research (Duan et al., 2019) explores, the automation of tasks by AI tools to reduce manual efforts, faster development

cycle and produce high-quality code. GitHub Copilot, Tabnine helps software developers by writing boilerplate code that reduces development time and avoids repetitive work. (Peras et al., 2023) describe that besides automation AI tools also generate valuable insight into the project to make decisions and create strategies. AI tools help developers fix critical issues first after prioritising the tasks based on project deadlines, potential risks and dependencies.

Enhance the Creativity of the Developers:

There isn't much proof that AI technologies can improve a developer's ingenuity by offering quick prototyping, insightful data, and innovative suggestions. (Barenkamp et al., 2020) contends that AI technologies like Kite and GitHub Copilot assist developers in showcasing their creativity by automating repetitive chores, including code development, debugging functions, testing, and reformatting. AI tools offer software developers a profound range of lucrative methodologies and way-outs, which in turn results in inspiring them to adopt the most worthy and actionable practice according to the desired need in order to improve their abilities for novel ideas in software development workflows (O'Leary, 2021). The below figure shows the different AI tools that are adopted as a key tool in software development processes.

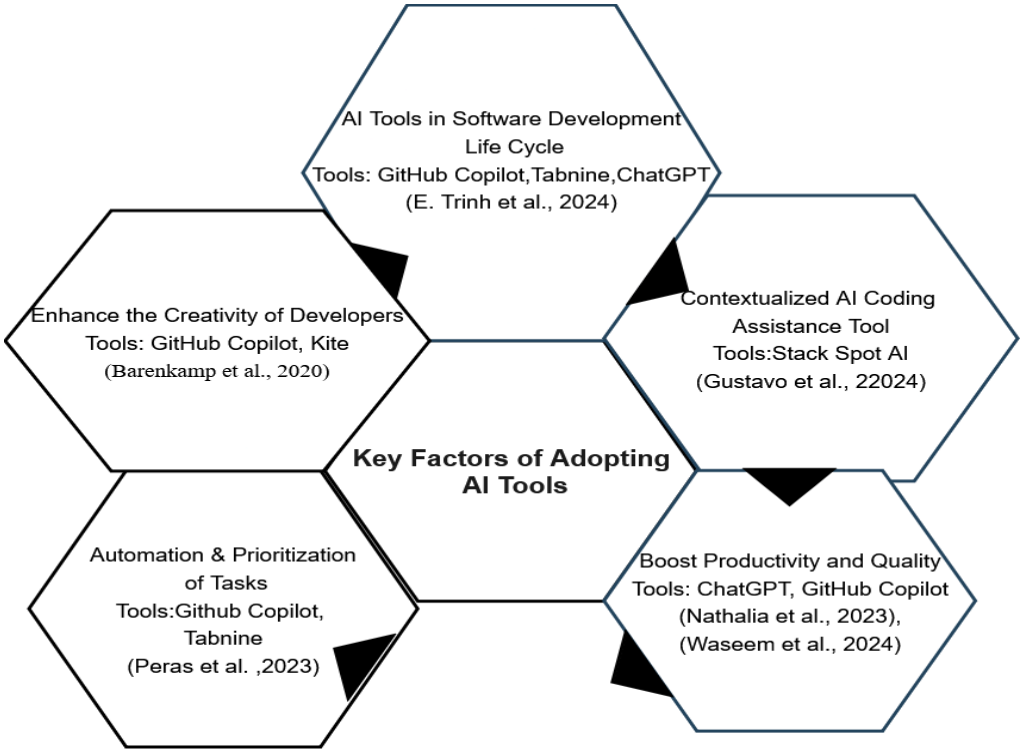

Figure 3: Highlighting Different AI Tools Adopted as a Key Tool

2.5.2 Benefits of Adopting AI Tools

AI Coding assistant tools are used to reduce human efforts and common mistakes that arise during the software development process by providing code suggestions, and code reviews. Nowadays, GitHub Copilot and ChatGPT are the most popular AI tools. Stack Overflow performed a survey in 2023 about AI popularity in software development and from 90K responses, 44% of respondents use AI tools in their software development premises (N. Davila et al., 2024). AI tools also benefit software development companies by speeding up the development process, creating better documentation and improving communication between team members (Barenkamp et al., 2020). This research discusses the major benefits perceived from the adoption of AI tools.

Useful Starting Point & Faster Development Process:

In software development premises, software developers find difficulties in starting their development works (Wendy et al., 2024) describe in her research that AI tools (i.e. GitHub Copilot) provide useful starting points for unfamiliar tasks. Besides AI tools also faster the development process so that developers spend more time on complex or unfamiliar tasks. The researcher illustrates that Generative AI ChatGPT provides initial insight to the developers while they are working with new/unfamiliar programming. It's a good learning opportunity for beginners. (Waseem et al., 2024), found out in their research that ChatGPT boosts their problem-solving skills and encourages innovative thinking to process faster development. In another research, Stack Spot AI is renowned for enhancing and simplifying the coding process. However, Stack Spot AI is a contextualized AI tool, is responsible for refining code, and quick command that helps the development process faster (Gustavo et al., 2024). GitHub Copilot speeds up the development process by converting comments into code and it can auto-fill the code to avoid repetitive tasks (N. Nguyen & S. Nadi, 2022). It can provide real-time code suggestions even it can generate functions and methods.

Improve Efficiency & Productivity:

AI tools are so extraordinary and bring a remarkable transformation in the software development workarounds. ChatGPT and Tabnine offer an amazingly effective range of means, to take the efficiency and productivity of software developers, to a whole new level,

in terms of code generation, debugging and documentation provision. AI tools are helping to remove skill barriers in software development. ChatGPT is also useful for inexperienced programmers because it provides valuable learning content, aids in skills development and develops better insights among programmers (Waseem et al., 2024). The researcher notes that utilizing ChatGPT and other AI applications boosts productivity by about 70%. It also assists in producing high-level demand specifications and detailed design specifications for small but complicated business applications (Asha R.,2024). ChatGPT provides a faster development process, it refrains developers from looking for solutions on the internet and explains the practical scenario and developers are able to learn in a short time and refine their focus on important aspects (Wendy et al., 2024). Other LLM tools, like Google BARD and Meta Code Llama, also reduce the time for code searching in other search engines to complete tasks and enhance the code quality, and as a result, these tools increase the productivity of developers and software quality (S. Sikand et al., 2024). AI-driven tools like Intellicode and Kite are contextual-based tools, and these tools use machine learning to provide contextualized code suggestions and auto-completion of code (Roland, 2021) to help the developers increase efficiency and reduce development time. StackSpot AI is also a contextual-based and conversational AI tool that provides more reliable code after understanding the context and delivering usable code snippets that increase productivity, and the developers can focus on more complex tasks (Gustavo et al., 2024). As an AI pair programmer, GitHub Copilot uses Natural Language Processing (NLP) to generate code, provide suggestions to write code, autocompletion code and increase the productivity of the pair human developers (N. Nguyen & S. Nadi, 2022). In one research, (Peng et al., 2023) describe that in a controlled environment, the group of software developers who accessed Copilot completed their tasks 55.8% faster than developers working without Copilot in pair programming.

Knowledge Exchange & Collaboration:

AI tools act as a significant platform which makes mutual knowledge sharing and communication much easier for software developers. These AI tools provide contextual insights, and automated information sharing that help developers to work more effectively to achieve the development goal (Saklamaeva, V. & Pavlič, L. 2024). Deepcode and Codacy are two mention-worthy tools which provide life-easing provisions for software developers,

by detecting issues in the code and conveying suggestions which are very worthy for new developers to improve themselves by aligning with the team's coding standard. GitHub Copilot provides contextualized knowledge and documentation to developers to access real-time information about a project. As an AI pair programming partner, GitHub Copilot cooperates with human developers in a real-time software development environment (N. Nguyen & S. Nadi, 2022). AI pair programming reduces developers' workload. As AI technology is evolving, it significantly changes the collaboration, communication and knowledge sharing facts among the team members to work effectively in a group.

Improve Code Quality & Reliability:

The AI coding assistant reduces development time and it integrates domain-specific knowledge (i.e. Stock Spot AI) to create more accurate code suggestions (i.e. ChatGPT, Tabnine) and significantly speeds up the development by automating repetitive coding tasks (Waseem et al., 2024), (Asha et al., 2024).ChatGPT instructs the way to enhance code readability and it also suggests code refactoring strategies to improve the code reliability. According to (Roland, 2021), it's not entirely true that AI tools generate reliable and high-quality code, however, AI tools provide suggestions for improvement and catch potential issues before they get bigger and affect the system. As a contextualized and pre-trained AI assistant tool, Stack Spot AI has great capabilities to enhance the quality of the code. It's true fact that Stack Spot AI is not a panacea, but it creates reliability while using its knowledge sources to their fullest to provide the best output. GitHub Copilot and ChatGPT have versatile abilities to reduce programming efforts, these tools can easily recall syntax and interact with the developers to provide appropriate code (S. Mohamed et al., 2024).

Enhancing Testing Capabilities:

AI tools significantly revolutionize the testing process. It helps to automate the testing cycle. As a popular AI tool, GitHub Copilot can generate test cases to automate the testing process. One study found that manually written test cases and test cases which are generated by GitHub Copilots are of the same value in terms of quality and effectiveness (S. Mehmood et al., 2023). AI-fabricated test cases can bring down testing time, cost and human endeavour. When a software developer encounters a bug ChatGPT generates a solution to identify and fix the bug and one research mentions that on the common test set, ChatGPT operates like

CODEX on a devoted deep learning (DL) based Automated Program Repair (APR) system and in this way, ChatGPT success rate is 77.5% (D. Sobania et al., 2023). In software development premises, developers identify bugs by analysing the source code and provide solutions by accelerating the debugging process. CLEVER is another risky commit detector that is based on ML to discover files that contain one or more bugs, and its accuracy is 75%. It is discovered by Ubisoft and Mozilla.

Code Suggestion, Code Review, Code Refactoring:

AI tools are competent to let us have context-aware code proposals that help optimize the performance of the coding and reduce error. (Gustavo et al., 2024) described in their research that StackSpot AI has a crucial feature in getting accuracy and completeness in code generation. It is capable of generating functional code snippets to accelerate the development cycle and reduce errors. However, Stack Spot AI is a contextualized-based AI tool and according to the project context, it can provide relevant, time-efficient and precise code suggestions. Its contextualized features help to generate accurate code and help developers to accelerate the development process. (Nathalia et al., 2023), reveals in an experiment that ChatGPT provides code suggestion/solutions in certain tasks that outperforms the novice programmer's performance in solving easy or medium-level problems. In another study, the researcher puts earnest effort into stating experimental observations, to make a definitive comment on correctness and whether code proposal by GitHub Copilot is intuitively understandable on 33 LeetCode questions in Java, JavaScript, Python, and C. The Copilot's code suggestion in Java was 57% correct whereas it is as low as 27% in JavaScript (N. Nguyen & S. Nadi, 2022). Tabnine is another code completion tool that uses Generative Pre-Trained Transformer 2(GPT-2) for deep learning models, and it can analyze the software developers' current code to foresee and propose the next line ((Roland, 2021).

DeepCode is backed by an ML-based code assessment tool which is schooled on thousands of open-source repositories. It is used for scanning codebases for bugs, performance factors and securities issues. It provides real-time feedback while using ML techniques to analyze code patterns (Roland, 2021). SonarCube is another code quality control that collects and analyses the source code, and it can generate reports about the code quality and standards of

the project. It uses the SQALE method to control the quality of the code. It's an amalgamation of two static code assessment tools which are known as Check Style and PMD. PMD is responsible for source code analysis whether Check Style is validating the code standard and finding the potential bug in the source code (Daniel et al., 2017). These tools have key contributions to maintaining the code quality, standards, and performance of the software products in the software development process. AI tools assist developers in refactoring the codes by identifying the unnecessary parts, inefficient and difficult to maintain. KITE is an AI code assistant refactoring ML tool that is trained on 25 million files. It suggests more effective and simple code. It can refactor the redundant code into a maintainable structure (Roland, 2021). WhatTheDIff is another code review optimization that simplifies the code review process, and it also provides instant code refactoring suggestions. These refactoring tools restructure the code to improve maintainability without changing its functionality.

2.5.3 Barriers to Adopting AI Tools

It's a fact that there is not enough research that AI tools have a great impact on software development, especially in automation and code quality. Government organizations, banks, and financial institutions are not integrating AI tools in their software creation process because AI tools are not gaining higher priority in the trust and security of use compared to conventional tools (Roland, 2021). While developers and organizations are adopting AI tools in their software development, they should be aware of the barriers to AI tools because tools can hinder the perfection, security, and effectiveness of AI tools.

Low Accuracy of Code:

As a generative AI ChatGPT is capable of generating code, however, all suggested codes do not have a good level of accuracy and are difficult to understand (Waseem et al., 2024). While improper or poor code is delivered by ChatGPT to the developers, it requires refining the code (Wendy et al., 2024) and causes an increase in the development time and cost. If the code suggestions are not up to date, it also affects the versioning of the programming languages. (J. W. Purwoko et al., 2023), ChatGPT may generate incorrect codes for

developers without understanding the deep concern of the project. This approach can lead to security vulnerabilities, incorrect code, or solutions. The researcher also mentioned that ChatGPT can't replace 100% of the software development work. One research explores Stack Spot AI as providing code suggestions as a generic solution to developers, and it is not context-based. It requires extra refinement to get the desired output (Gustavo P., 2024). According to (S. Imai, 2022), in pair programming, GitHub Copilot generates more lines of code than its human pair developers; however, the code quality is lower.

Inconsistent Response & Trust in AI Tools:

AI tools are trained on vast amounts of datasets that consist of various programming patterns and techniques. It's difficult to confirm that the dataset covers all the aspects of coding needed for developers. As (Gustavo et al., 2024) mentioned, Stack Spot AI provides varying responses for an identical issue. Its response creates confusion among developers, and the reliability of the tool is questionable for repetitive tasks. In another research, (S. Imai, 2022) describes that as a contextualized-based AI tool, GitHub Copilot can generate inconsistent responses and create boilerplate code that works for standard situations but is not well suited for specific contexts, and it may lose the trust of the developers because of missing components, errors, and security vulnerabilities. (Roland, 2021) describes that AI tools should be properly documented as a USSR manual. Without proper documentation, the developers will not be able to understand the features and use process of AI tools, and it is one of the significant barriers to adopting AI tools in software development, AI tools also lose their trust.

Diverse Source of Knowledge:

AI tools generate inappropriate code suggestions for software developers when the tools do not have access to domain-specific knowledge or business rules of certain software products. ChatGPT is an LLM-based AI tool that uses diverse sources of knowledge to generate responses; however, it is difficult to obtain accurate results, and it may hinder the developer's trust in the tool (Gustavo et al., 2024). In another research, ChatGPT uses LLM and diverse sources of knowledge for specific tasks; its consistency is about 30% in a generated code, and numerous inconsistencies in code required developers' manual efforts (D.-K. Kim et al., 2023). (Gustavo et al., 2024) illustrates that Stack Spot AI can access various sources of

knowledge, and it is beneficial for software developers; if the sources are not accurate or relevant, it leads to generating inappropriate code suggestions for software developers. GitHub Copilot is also trained on extensive knowledge, and the developers do not know the internal model of the GitHub Copilot and are not sure sure the exact training data of this tool, so it is questionable that exact queries exist in the current dataset (N. Nguyen & S. Nadi, 2022). It is difficult to predict the outcome is reliable for the developers.

Inexperience Developer & Complex Task:

Adopting AI tools can be beneficial and challenging for both experienced and inexperienced software developers. AI tools assist developers in automating repetitive tasks, code suggestions, and testing; however, there is no evidence that AI tools like ChatGPT, GitHub Copilot, Tabnine, and Stack Spot AI help inexperienced developers bridge the knowledge and skill gaps for complex tasks. One study describes that ChatGPT may help novice developers to solve simple tasks, but there is no evidence that ChatGPT outperforms experienced developers for complex tasks (Nathalia et al., 2023). One researcher explores that ChatGPT has generative capabilities, sometimes it requires subject matter experts to validate the generated code when the context is complex (Asha et al., 2024). Stack Spot AI is good for basic code generation; however, this tool may struggle with more complex coding requirements, and after closing a conversation with this AI tool, the history can't be recalled again. This particular problem hampers the complex tasks, extends the development time period, and increases the development cost. In the absence of adequate plugin features, certain functionalities in Stack Spot AI for IDEs restrict the usability of Stack Spot AI in the software development process (Gustavo et al., 2024). Over-reliance on AI tools hinders the learning process of the software developers, such as critical thinking and debugging skills, and results in poor code quality and security vulnerabilities in the software.

Interruption & Security Concerns:

AI assistant tools GitHub Copilot leads to distraction in software development by providing continuous suggestions to the software developers. GitHub Copilot continuously generates code suggestions when the developers wish to work without any interruption and the developers feel uncomfortable in this circumstance. In another situation, the developers also feel bothered when the context is precise, but the GitHub Copilot suggestion is a large

volume of code at the start of the code creation process (Wendy et al., 2024). There is not enough research on how AI tools are responsible for generating security and privacy concerns issues. Copilot completes with a human pair: a novice developer in coding, and it's difficult for a novice developer to find bugs and refactor complex code, and thus it affects the security and quality of the software (D.-K. Kim et al., 2023). ChatGPT is an LLM-based generative AI that generates codes without properly knowing the context of the project. It creates security vulnerabilities and increases the inaccuracy rate in the production environment (J. W. Purwoko et al., 2023).

2.6 AI Tools in Scrum and Sustainable Software Development

Artificial intelligence (AI) has the potential to alter dramatically project management by facilitating intelligent recommendations and the automation of operations and project analytics. However, there is limited empirical evidence that Agile methodologies like scrum is particularly benefit from AI tools. However, AI tools to facilitate task prioritization, more precise estimates, and better backlog management across themselves are available in Scrum. AI tools help Scrum teams to stop applying time to low-level problem-solving and mundane tasks by automating those tasks and providing real-time insights. According to (M. Kanbur et al., 2023), there are various types of AI tools are used in project management and many of the best fit for scrum management such as Aconex, Fluid, Clarizen, Huginn, Jira, Monday.com, Smartsheet, Nifty.ai, Wrike etc. Aconex and Fluid are used to automate tasks, improve interaction between team colleagues and track the progress of the sprint in Scrum. Besides, Clarizen and Huginn are also used to automate tasks such as scheduling reports and generating reports. Monday.com is a software project management (SPM) tool, that guides the team members to stay organized and keep on track. Jira has various AI features such as predictive analytics and machine learning. It boosts team efficiency discovers knowledge and accelerates decisions. Nifty.ai and Wrike are both AI tools responsible for planning, tracking, and managing their projects. These AI tools may have a great impact on scrum process management. The table 2 shows the different AI tools and their impact on the scrum process.

Table 2: Impact of AI Tools on the Scrum Process (M. Kanbur et al., 2023)

Scrum Process	**Impact of AI**	**AI Tools**
Sprint Planning	Prioritize tasks, Estimate time and resources to complete tasks	Smartsheet, Clarizen, Huginn
Daily Stand Up	Track team progress, identify Blockers	Monday.com, Aconex, Fluid
Epic	Breakdown Epic into manageable smaller chunks	Smartsheet, Jira
Work Breakdown	Create a detailed plan for each task that includes dependencies, resources	Nifty.ai, Wrike
Backlog	Manage product Backlog, prioritise task,estimate time and resources to complete tasks	Clarizen,Huginn, Smartsheet
Product Leadership	Help to make better decisions about product roadmap, features and user experience	Jira,Nifty.ai, Wrike

However, (M. Kanbur et al., 2023) showed that AI tools can make task management processes more streamlined, improve sprint planning, and boost team collaboration, ultimately increasing the efficiency of development processes. He also describes that using AI to drive task prioritization also reduced decision-making time in sprint review and, in general, improved team productivity.

Software sustainability is emerging to develop flexible digital ecosystems and aim to minimize environmental impact, promote greener systems, and optimize resource usage. To bring sustainability to software development premises, sustainable coding practices are introduced. Nowadays, software developers are using AI tools to generate code in software development projects. However, there is a lack of research papers showing that developers are using AI tools to generate code and follow sustainable coding practices (S. Sikand et al., 2024). Several studies suggest that 2-7% of greenhouse gas emissions occur because of software and corresponding hardware (Freitag et al., 2021). In software development, developers don't optimize code in terms of sustainability, and it is responsible for high carbon emissions (Gustavo et al., 2017). Developers are not aware of sustainable coding practices, and they use energy-consuming patterns instead of energy-efficient ones. The using of AI tools in software creation is getting more popular day by day. There are no notable studies that evaluate the sustainability aspects of generated code. However, without concerning sustainable aspects, some studies focus on security, correctness, quality, maintainability, and performance (R. Khoury et al., 2023). (S. Sikand et al, 2024) performed a deep analysis in their research to evaluate the sustainability aspect of AI-generated code by ChatGPT, Copilot, and BARD. The researchers chose 5 green rules developed by CAST and consist of energy-efficient code patterns. However, in this research, the researcher reveals in certain rules Copilot generated code is 40% non-green and ChatGPT is not able to generate green in 33% cases for sustainable practices as well as BARD. ChatGPT, Copilot, BARD training method, and attributes are following the sustainable practice so in most cases these are not able to produce green code and have a smaller impact on environmental sustainability.

According to the Karlskrona Manifesto for sustainability, it has five dimensions: environmental, technical, individual, social, and economic dimension (P. Bambazek at el., 2022). AI tools may have greater impacts in both the development process and sustainable

impact. AI tools (i.e., GitHub Copilot, Tabnine) automate tasks to reduce manual efforts, have a faster development process, and produce high-quality code. AI tools are responsible for reducing operational costs by automating repetitive tasks like code generation and bug fixing (Duan et al., 2019) and generating boilerplate code. Overall, it is responsible for reducing manual labour, using resources more carefully, and reducing development costs, and GitHub Copilot can generate test cases to automate the testing process (S. Mehmood et al., 2023). AI-driven test case generation is reducing testing time, cost, and human efforts, and the detection of bugs improves the software quality, reduces economic waste, and contributes to economic dimensions. AI tools help developers enhance their individual skills and productivity. Tabnine and ChatGPT help developers to enhance their skills and remove skill barriers by generating code, and documentation facilities, and aiding novice programmers (Waseem et al., 2024). In research, (Wendy et al., 2024) explained that ChatGPT provides faster suggestions and explains the real-life solution to the developers and refrains from accessing the other search engines, and thus developers can engage in more critical tasks and innovative development to improve individual productivity.

2.7 Research Gap & Problem Statement

In the present time, AI tools are being deliberately adopted by software developers to optimize their tasks, to ensure optimized output and ease their professional life, for instance: code review and bug detection in the scrum process. Few research indicate that software developers are very concerned about security procedures and issues while AI tools such as Copilot, and ChatGPT generate codes, and detect bugs (Klemmer & Jan, 2024). Other studies reveal the benefits of adopting AI tools in the working environment without concerning the reliability of the generated codes, bug detections or ethical matters (Roland, 2021). However, there is limited research that describes the necessity of AI tools in the scrum process on software developers and how they contribute to both sustainability and software products. Only a few research papers investigate the benefits and barriers of AI tools usage in the process of creating software. Still, these research papers don't investigate how these AI tools increase the productivity of software developers in their working premises. AI tools help developers to mitigate their development tasks, code generation and review, bug

detection and repair the program. Yet there is no clear data that AI tools have a broader impact on the scrum process and nor have a greater contribution in sustainable impact in software development.

This study put an earnest effort, to find out the most alluring and deciding factors, of these AI tools, which software developers yearn for. In a scrum-based project, how do these AI tools benefit the software developers, and ease their software development process, these obstacles arise while using AI tools, key facts that motivate the developers to adopt the AI tools compared to traditional software development tools respectively. This study also aimed, How AI tools significantly increase the productivity of software developers from development to software maintenance and the contributions of the developers in sustainability so that the software can be used in the long run and easily maintainable.

3 METHODOLOGY

This section illustrates the systematic approaches to providing the key factors, benefits, and barriers to adopting AI tools in the software development workflow by software developers. This will also focus on exploring the potential of AI tools to promote sustainable practices among software developers. This methodology section is designed to explore the broad understanding of existing literature about AI tools in software development premises and in-depth insights from the software developers about sustainable developments by using AI tools.

3.1 Research Method

In this section, the research employs a mixed-method approach (Johnson & Onwuegbuzie, 2007) to discuss the impact of AI tools that are adopted in the software development process, especially in the scrum by the software developers and explores the insights into how these AI tools increase the productivity of the software developers and contribute to sustainability

in software development. The method is divided into two phases: the Systematic Literature Review and the Semi-Structured interview process. Each stage is related to the specific research question and method to achieve the goals of the research. This approach integrates, both interpreting perceptions from SLR and capturing real-time insights of software developers through a Semi-Structured interview process to get the validated and combined output from this research.

To answer the first research question, this study follows the Systematic Literature Review (SLR). SLR informs readers on the state of the literature on a given topic by using a systematic approach of gathering, determining, and critically analysing the existing research's identifies key information about a subject and recommends topics for further research. (Angela et al., 2022). For this research, SLR identifies existing research that helps to find out crucial factors in the use of AI tools in the software development process. These existing researches also explains how these AI tools contribute to software developers' workflows, especially in code generation, code review, and program repair and discuss the benefits, and barriers of using AI tools in the software development process compared to traditional tools. The output of this research is discussed in section 4.1

To answer the second research question, a semi-structured interview process is arranged between software developers who have worked in different software development companies, especially in scrum projects and use AI tools for development purposes. The interview explores the insights of the software developers on how AI tools significantly improve their skills and productivity, especially in code generation/optimisation, debugging, and bug detection. Through the interview process, this research is also trying to capture the view of software developers on how AI tools reduce time and effort in the software development process compared to traditional software development tools. Besides, this phase focuses on how AI tools help the developers contribute to gaining sustainability in the software development process specifically in code quality, bug repairing, performance optimization, and cost minimization of development. The result of this research is explained in section 4.2

In the end, this research reveals the key findings of the impact of AI tools that increase the productivity of software developers in the creation of software products and the contribution

of software developers in sustainability by reducing software development time, and cost; maintainable, durable and stable code. This study pinpoints the factors that help developers to improve their productivity and contribute to sustainability. These perceptions are elaborated in sections 5.1 and 5.2.

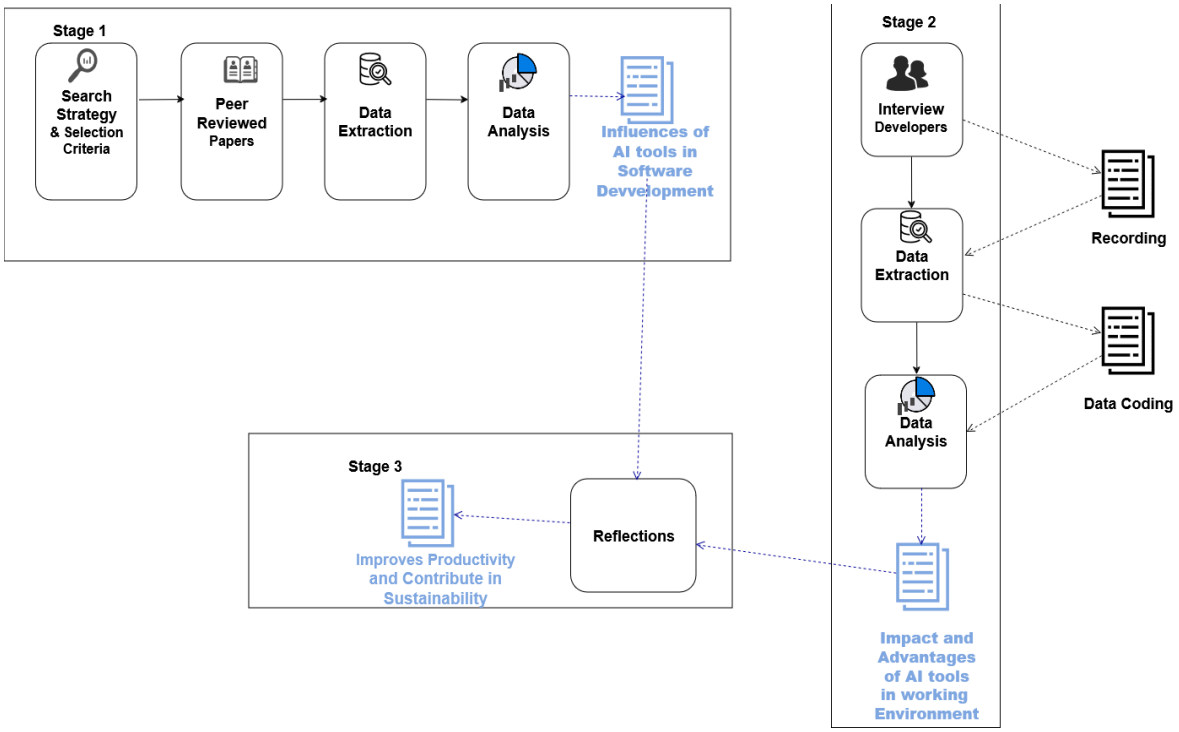

Figure 4: The Research Process Diagram

3.2 Systematic Literature Review

The literature review is the basement of all research. A literature review often offers an overview, synthesis, and critical assessment of previous research articles, looks at and assesses the state of knowledge, and identifies and develops new research issues along with relevant research topics. (Boell et al., 2015). Though there are different types of literature review processes, this research is conducted through the systematic literature review process as described (Moher et. Al, 2009). It describes the access to databases, searching strategies, selection criteria of the papers and the analysing process of the literature.

3.2.1 Search Strategy

The search strategy is an important component of the systematic literature review. It's the basement of identifying and evaluating relevant research papers. This strategy set up the search parameters that include ideas, databases, and inclusion/exclusion criteria to gather a variety of research papers without being biased. The primary sources of the literature are the Web of Science, ACM Digital Library, IEEE, ScienceDirect, Scopus, and ProQuest digital libraries. The search strategy is a combination of keywords and Boolean operators to build search queries to find the relevant papers and the queries are

("AI Tools OR Artificial Intelligence OR AI) AND ("Software OR Software Development") AND ("Benefits OR Advantages")

("AI Tools OR Artificial Intelligence OR AI) AND ("Software OR Software Development") AND ("Software Developer OR Software Engineer")

3.2.2 Selection Criteria

The selection criteria determine which research papers should be included in the literature review. This criterion ensures the most relevant and quality research for a focused analysis. The selection procedure of the research papers is divided into inclusion and exclusion criteria. To select the relevant research paper. Each type is used to filter papers based on parameters that are related to research questions.

Table 3: Inclusion and Exclusion Criteria for Selecting Studies

Inclusion Criteria	**Exclusion Criteria**
Papers are written in English	Papers not in English
Papers Published from 2019 to Present	Not peer-reviewed papers

Directly address the impact of AI tools on the software development process	Vaguely focus on AI tools in the context of software development
Studies that relate software developers, engineers, or related professionals in the context of software development and AI tools	Focus on AI applications outside of software development, such as in other industries (e.g., healthcare, finance)
Papers that focus on the usage of AI tools in software development and their impact on sustainability	Papers that have unclear methodologies, or insufficiently detailed findings

At first, this research extracted a total number of 2148 papers pulled out from digital libraries including ScienceDirect, Web of Science, IEEE, ACM, and Scopus. After identifying the initial papers, duplicated papers are removed. The round of exclusion is based on the title of the papers, screening the abstract of the papers, papers not related to AI, software development, and papers vaguely related to AI. Finally, 24 papers are selected, where a total 20 number of papers are selected after inclusion and exclusion and 4 of them are selected from snowballing. Backward snowballing refers to the relevant research papers that are not found in the initial database searches but are found in the reference list of the selected papers for review (S. Jalali & C. Wohlin, 2012). Here is the visual representation of the selection

criteria.

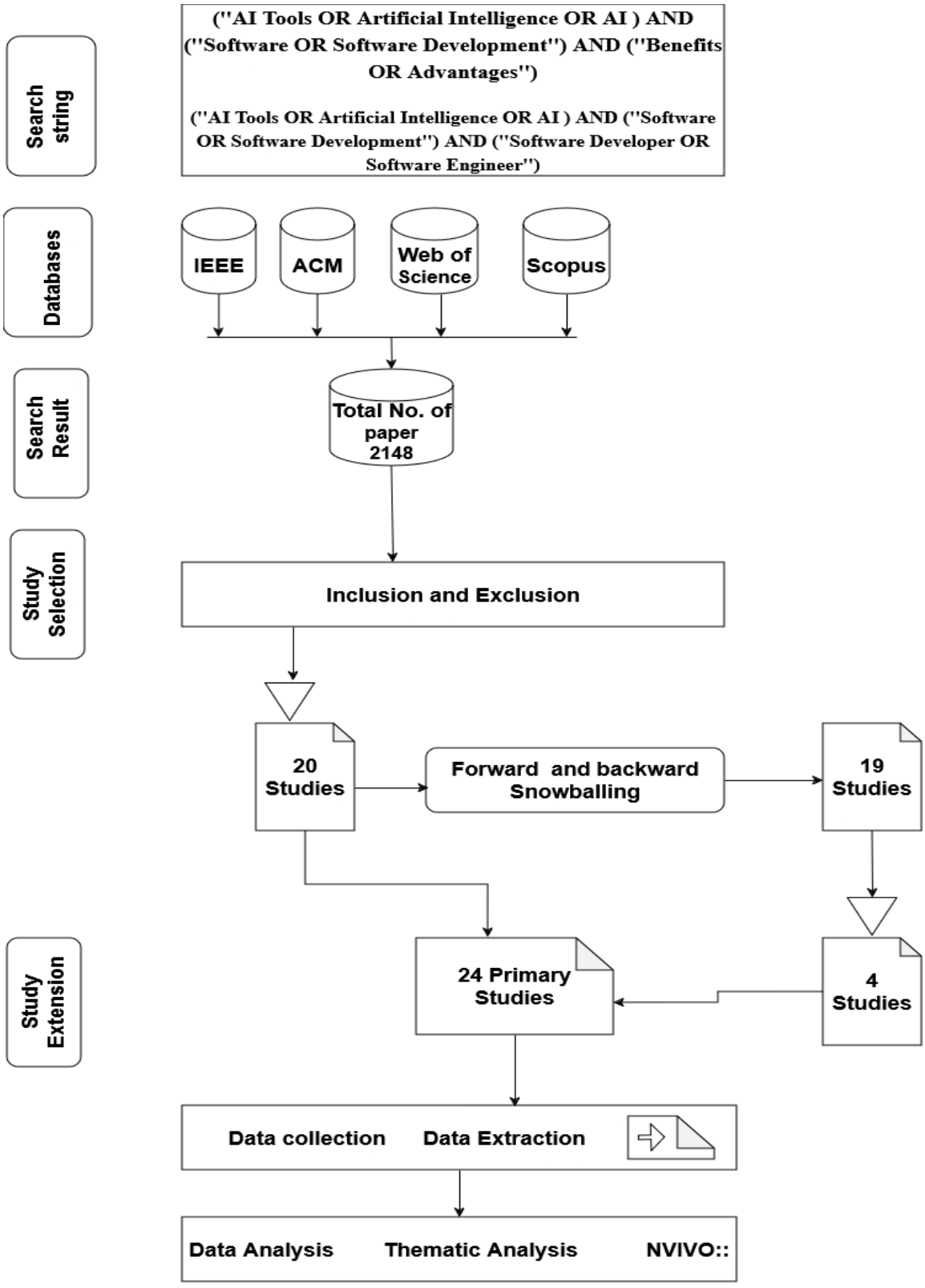

Figure 5: Literature Review Process

3.2.3 Data Extraction

In a systematic review, the data extraction process is a crucial step to identify the research papers through the search technique. The process incorporates in-depth analysis carefully pulling out information from all the selected studies that best relate to the research questions for further synthesis. In order to have in-depth knowledge of the objectives of all the real research papers and their goals from a broad perspective, it is crucial to start the process of research with the analysis of the abstract and introduction of each paper. The study explores and examines the research problem, situating it within the context of embracing AI tools in the creation of software. Afterwards, it evaluates the approach to determine the reliability and accuracy of the results.

A Google Excel sheet is utilized to gather data systematically from each of the selected studies to collect actual data. This sheet is designed to capture both general and specific information such as author name, title of the study, year of publication and specific data such as name of AI tools, key factors, benefits and barriers for adopting AI tools in software development.

3.2.4 Data Analysis

Thematic analysis is used to analyse the qualitative data that are extracted from research papers. This process helps to identify data, categorize data, and process data to develop a pattern or theme that motivates the development of results within the research (Vaismoradi, et al., 2016). The first step is to read the research papers to familiarize them with the data. According to the research questions, the significant sections are marked to generate the initial code. Similar types of codes are being grouped to develop a theme that bears the potential meaning of the data set. This research is concerned with reviewing the developed themes and finalizing the themes after screening of the reflected data and code. Finally, the theme is defined by the name that represents the actual data set. This research uses a Google Excel sheet to complete the data analysis process.

3.3 Interview Design

The semi-structured interview is designed in this research in a systematic approach to gather comprehensive qualitative data (S. E. Hove and B. Anda, 2005) from the software developers who are currently working in software development and availing AI tools in the software development workflow. The semi-structured interview procedure contributes to an inevitable role in extracting the purport of the software developers to reveal the factors of adopting AI tools and explain how AI tools help developers to increase their productivity and their contributions towards sustainability.

3.3.1 Recruitment Process & Participants

Recruitment of participants is one of the crucial steps to identify and involve the right participants who are working in the software domain and have knowledge to provide meaningful insights for the research. This research follows inclusion and exclusion criteria to select participants for the interview process.

Table 4: Recruitment Process for Selecting Participants

Inclusion Criteria	**Exclusion Criteria**
Good command in English	Poor Command in English
Member of a Scrum Team	Not a member of the Scrum Team
Software Developer	Not a Software Developer i.e Tester, System Administrator, QA
Must have Experience in software development for 4 years	Novice developer

Use AI tools for software Development	Use Traditional Tools for software development
Minimum idea about sustainability in software development	No idea about sustainability in software development

The participants are selected from different regions around the globe. Before selecting an interview, participant's LinkedIn profile is thoroughly observed to contribute to this research. Initially, a total of 27 participants were selected for the interview. After applying inclusion and exclusion criteria, finally, 16 participants were selected for the interview session and emailed for their permission to set up a meeting. Among them, 11 participants agreed to join in the interview session. Every participant is experienced in the software industry and works as a team member in a scrum in reputed software companies. There is a list of participants:

Table 5: List of participants selected for Interview

Participant	**Company Name**	**Country**	**Role**	**No. of years Working in the Software Industry**	**No. of years working on Scrum Projects**	**Projects worked on**
P1	Cefalo Bangladesh Ltd.	Bangladesh	Lead software engineer	13+	4+	Web applications, Mobile application
P2	Bitsmedia Pte. Ltd, Singapore	Singapore	Senior Software Engineer	12	8	Mobile application development

P3	ePrice Oy	United Kingdom	Software Engineer	10	8	Web Applications and Enterprise Software
P4	Reputed German Bank	Germany	Software Engineer	8	8	Enterprise Software
P5	Central Provident Fund Board, Singapore	Singapore	Software Engineer (Java)	11+	11+	Web applications, Enterprise Sol.
P6	IdeaScale	Bangladesh	Senior Software Engineer	12	10	Web application, Enterprise Software
P7	Treon Oy	Finland	Software Design Engineer	4	4	Web application, Enterprise Software
P8	Tuum	Estonia	Senior Software Engineer	12	12	Mobile Application, Enterprise software (Spring Boot)
P9	Rakuten Group Inc.	Japan	Backend Developer	6.5	3	web applications, enterprise software), Backend development, Mobile and Web Applications
P10	Data path	Bangladesh	Lead Software Engineer	14	12	Web and Desktop app, Cross platform mobile app and Enterprise application
P11	Pathao Ltd.	Bangladesh	Senior Software	6	6	mobile apps, enterprise

			Engineer(iOS)			software) Mobile Application(iOS), Web Applications.

3.3.2 Interview Format

The interview is conducted following the semi-structured interview method that offers a balance after a fixed set of questions and then carries forward to explore more as arises in the course of the discussion. The interview sessions take place through Google Meet conferencing, and it is easy for the participants to immerse themselves in the conversation from different parts of the world. These sessions are recorded as audio conversations after participants agree to record.

3.3.3 Interview Questions & Guideline

The interview sessions are conducted using a set of previously designed questions that were related to research questions. These questions are designed to ensure that all the key points of the research are fully covered and result in meaningful insights. The interview process is guided by (Cohen et al., 2017), which allows flexibility for the interviewers to go into depth insights to explore interesting topics around the conversation. This approach helps to conduct interviews in a consistent way and collect data in a systematic way.

3.3.4 Data Collection & Recording Process

The interviews are conducted after the participants agree to both interviews and recordings. At first, the goal of the interview is explained to the interviewee, and the interviewee is asked the questions one by one to ensure that the interviewee can contribute confidently to the research. Each interview ranged from 30 to 50 minutes in duration so that the interviewee could get enough time for an in-depth response and discussion. The records are transcribed

automatically using the automatic transcription tool Bluedot. Afterwards, manual corrections are made to the transcript to ensure the accuracy of the data.

3.3.5 Interview Analysis

This research includes a thematic analysis process to analyse data from the interview data. This process helps researchers to analyse data in qualitative research and get insights from the data to discover meaningful knowledge. To develop patterns or themes from the data, it follows some steps to identify data, categorise data, and process data in a meaningful way (Vaismoradi et al., 2016). NVivo 14 is used to analyse the data from the interview transcript. This software is very popular by researchers for qualitative research. It facilitates the researchers to organize data analyse them and visualize data from interview or survey processes to discover the meaning of insights. The thematic process involves a series of activities: to find out the initial code, the important sections or phrases are highlighted in the transcript. To have meaningful insight from the data set, similar types of codes are grouped to build a theme. This research is concerned with reviewing the developed themes and finalizing the themes after screening the reflected data, and code. Finally, the theme is defined by the name that represents the actual data set.

4 RESULTS

This chapter outlines the key findings from the research according to the research questions in this study. 4.1 section is dedicated to presenting the benefits and barriers of adopting AI in the software creation process. This research conducts the SLR process to find the benefits and barriers of using AI tools in creating software. 4.2 presents the insights of software developers over the advantages of AI tools in software development premises and the impact of AI tools on sustainability.

4.1 Research Question 1 (SLR)

To answer RQ1, the SLR is followed to reveal the current state of knowledge from research papers to identify the benefits and barriers of using AI tools in the creation of software products. The SLR follows the predefined process mentioned in Section 3.3 to ensure the proper outcomes. Finally, 24 papers are elected to analyse data from the literature.

4.1.1 Key Findings From SLR

This study develops valuable key themes and perceptions to answer RQ1 related to the benefits and barriers of adopting AI tools in the process of software creation. This research discloses several themes to identify the benefits and barriers of AI tools and these crucial themes are Benefits of AI and Efficiency Boosters(focusing on Productivity, Code Quality, Cognitive load, testing), AI as an Augmentative Tool for Routine Coding (Support for Routine Tasks, Useful for Novices), Challenges and Constraints in AI Adoption (looks for Low Accuracy and Consistency, Requirement for Supervision, Limited Complexity Handling), Reliability and Security concerns (provide importance on Inconsistent Task Understanding, Security Concerns, Dependence on Model Version). This segment illustrates key features of the developed themes to generate valuable findings for RQ1. Figure 6 illustrates the identified benefits and barriers of adopting AI tools in the building of software.

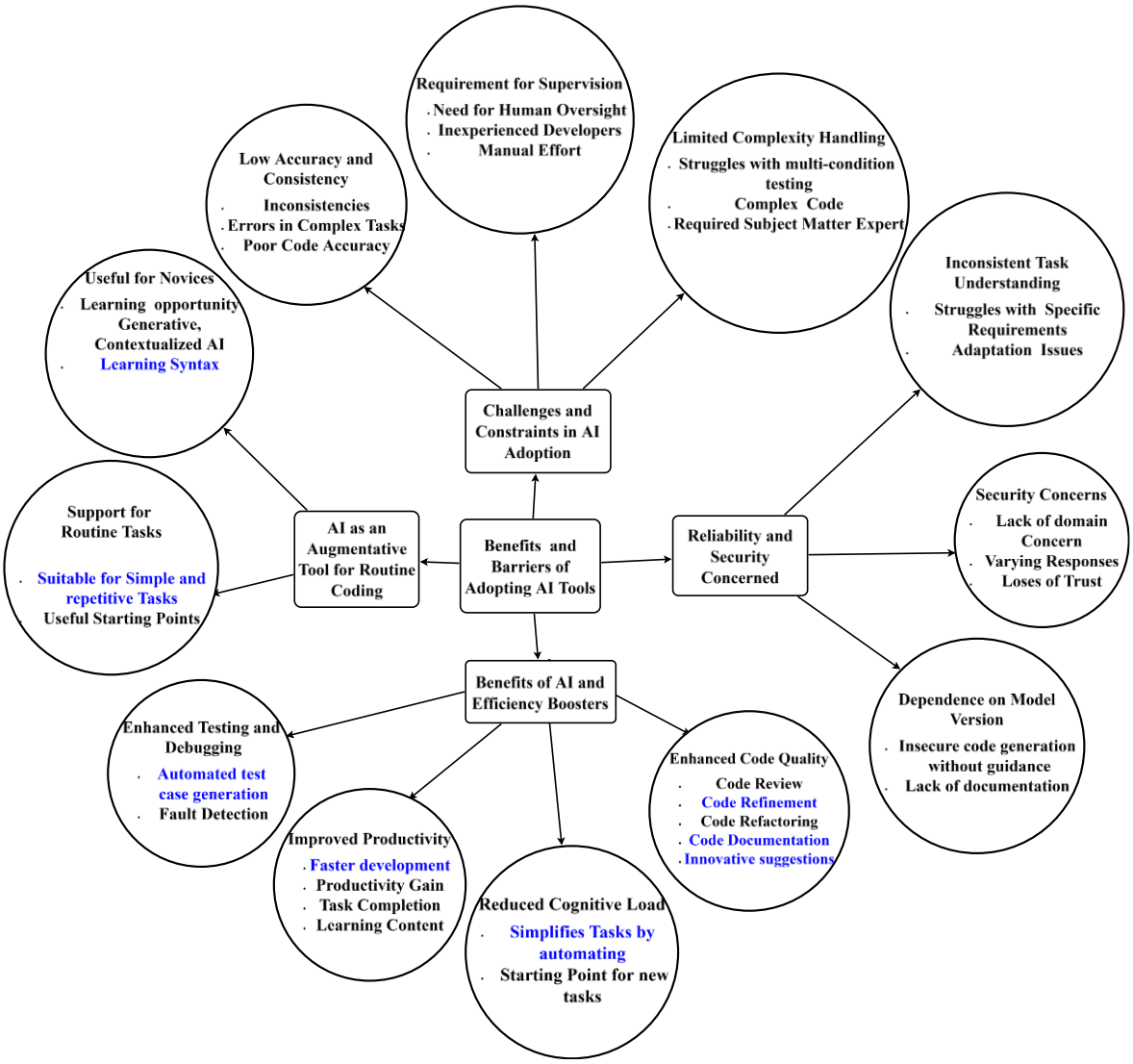

Figure 6: Identification of Benefits and Barriers of Adopting AI tools from SLR, see Appendix A for related reference

Theme 1: Benefits of AI and Efficiency Boosters:

This Theme discusses the benefits of adopting AI tools in the software creation site and how these AI tools boost the efficiency of the software creation process. As a Generative AI ChatGPT improves productivity through faster development (Wendy et al., 2024) in different SDLC phases it gains productivity by almost 70% in SDLC phases (Asha et al., 2024). AI tools also create learning opportunities through the creation of software products such as conversational assistants (Waseem et al., 2024). In a controlled environment, GitHub Copliot is able to complete tasks at a rate of 55% (Gustavo et al., 2024).

AI coding assistant tools are also helping software developers by enhancing code quality through accurate code, code review, Code refinement, Code documentation, Code refactoring, and Innovative suggestions. GitHub Copilot as a contextualized AI tool provides more accurate code. Code review helps to identify potential bugs and ChatGPT improves the syntax by recalling and interacting with developers (S. Mohamed et al., 2024), code documentation makes the code to reuse for further use. AI tools also provide innovative suggestions that boost the learning opportunities of software developers (Peras et al., 2023), (Waseem et al., 2024).

Github Copilot smooths the development process by reducing the cognitive load of the developers. It provides a useful starting point for the unfamiliar task Wendy et al., 2024). For that reason, the developers don't need to access Google, Stack Overflow or any other search engine for a solution. It saves development time and cost. These tools also simplify tasks, and it help novice programmers. AI tools also accelerate the testing process. ChatGPT has a reputation for identifying and fixing bugs and in the common test set its success rate is 77.5% (D. Sobania et al., 2023). GitHub Copilot can generate automated test cases to make the testing process faster and detect faults, however, refinement is needed for complex tasks (S. Mohamed et al, 2024).

Theme 2: AI as an Augmentative Tool for Routine Coding

This theme aims to intensify the software development process through routine tasks. It can auto-suggestions and completion of code to avoid repetitive tasks. It simplifies the routine coding tasks and works as a helping hand in fast-moving workflows. GitHub Copilot and ChatGPT have versatile abilities to reduce programming efforts and interact with the developers to provide appropriate code (S. Mohamed et al, 2024).

GitHub Copilot, Tabnine, and Stack Spot AI help inexperienced developers bridge the knowledge and skill gaps for complex tasks. As an AI pair programming partner, Github Copilot cooperates with human developers in a real-time software development environment (Nguyen, 2024). So, the novice programmer does not need to rely on others.

Theme 3: Challenges and Constraints in AI Tools Adoption

This theme elaborates on the challenges of using AI tools and defines the factors that create constraints for adopting AI tools in the software development premises. As a generative AI, ChatGPT-generated codes are not consistent, and all suggestions do not have a good level of accuracy, and inconsistent responses affect reliability in the development process (Waseem et al., 2024). AI tools also deliver poor, improper code that needs to be refined again because of the complexity of the task (Wendy et al., 2024). StackSpot AI is also responsible for creating varying responses that cause confusion among developers (Gustavo et al., 2024).

AI tools are not capable of solving complex issues by themselves. These Tools require human supervision to solve problems. AI tools have generative capabilities, and it can provide solutions by generating code, however, it requires humans to validate the generated code (Asha et al., 2024). Experienced Developers are capable of understanding the solution provided by AI tools, however, inexperienced developers are not able to handle the code that is generated by AI tools. To maintain the code quality human supervision is required especially for novice developers. Stock Spot AI is really good for basic code generation, as conversational AI it can't regain the history of conversation, it can't handle complex tasks, raises the development time and cost and requires extra manual efforts to solve complex problems. AI tools such as GitHub Copilot is also struggling in solving the multi conditional testing because of their limited capabilities in handling complex scenarios (Gustavo et al., 2024). It requires subject matter experts to check the issue in the generated code (Asha et al., 2024).

Theme 4: Reliability and Security Concerned

This theme escalates the reliability and security concerns of AI tools in the software creation process. ChatGPT struggles to understand the specific coding requirements, so it can't generate reliable code and raises tension in security. Without Domain knowledge, ChatGPT's response can vary and generate insecure code (J. W. Purwoko et al., 2023) creating security vulnerabilities within the project and losing the trust of software developers (Gustavo et al., 2024). GitHub Copilot can generate inconsistent responses because of missing components, errors, and security vulnerabilities (S. Imai ,2022). ChatGPT is not able to handle complex coding tasks and without maintaining proper documentation, there

is a problem with maintaining versioning and it is difficult to compare codes in different versions.

4.2 Research Question 2 (Interview-based)

To answer RQ2, this study conducted a semi-structured interview process with software developers around different geographic locations who work in a different reputed company in Scrum. This section tries to reveal the insights of the software developers who adopt AI tools in their workflows to get the advantages of AI tools and their contribution to sustainability. The interview process follows the predefined process mentioned in Section 3.4 to the insights of the software developers.

4.2.1 Key Findings from Interview

This study generates important key themes and insights to answer RQ2 related to the advantage of embracing AI tools by software developers in their software creation process and their contribution to sustainability. This research releases several themes to the advantages of AI tools in software developers' workflow and sustainability and these important themes are Benefits of AI and Efficiency Boosters(focusing on Productivity, Code Quality, Cognitive load, testing), AI as an Augmentative Tool for Routine Coding (Support for Routine Tasks, Useful for Novices), Sustainable Development Through Efficiency and Automation (Reduce Development Time, Personal Growth, Minimizing Cost), Optimized Workflow and Long-term Code Quality (Reduced Redundancy in Development, Code Quality and Maintainability, Ongoing Code Reliability). Figure 7 illustrates the identified advantages of adopting AI tools in the software developers' working premises and their contribution to sustainability.

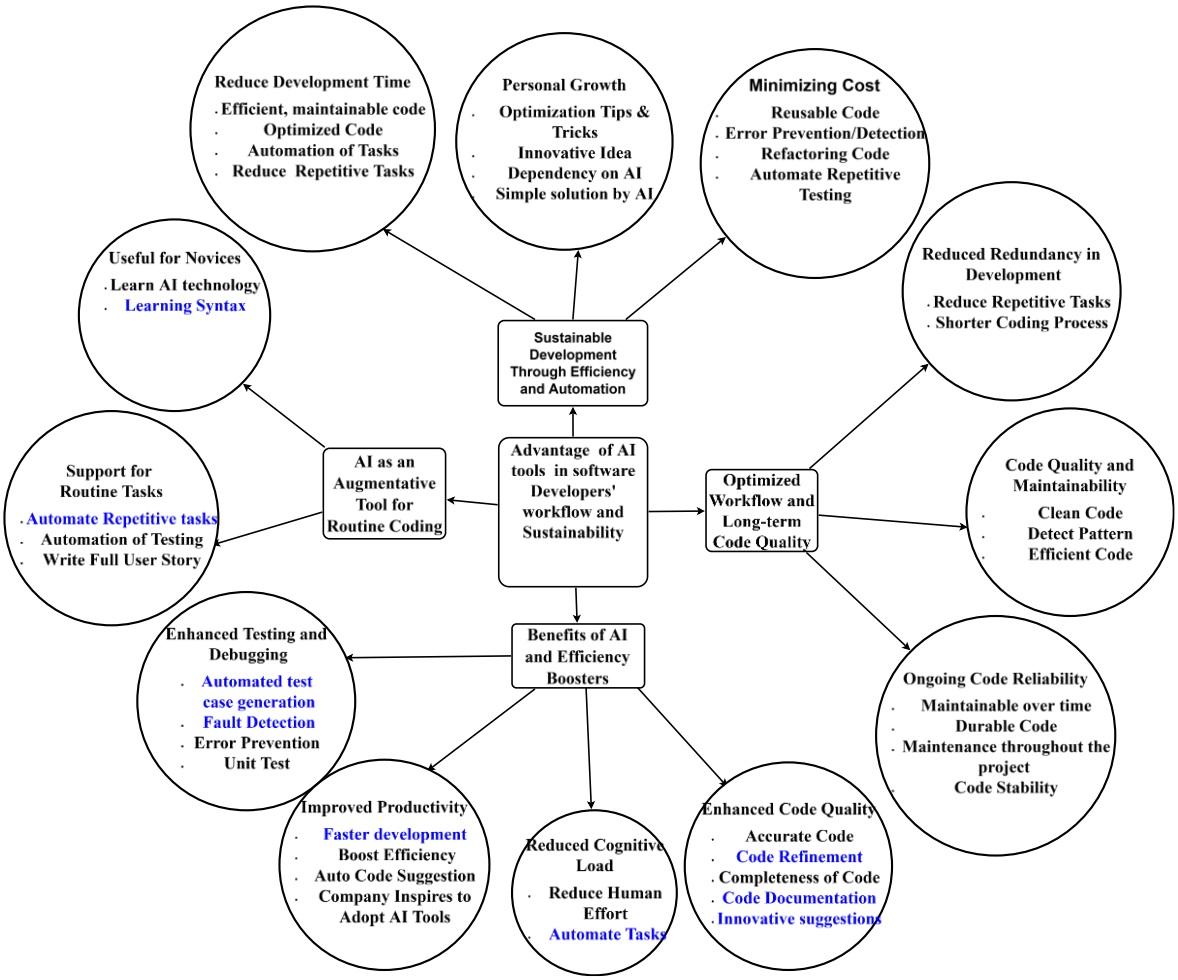

Figure 7: Identification of Advantages of AI tools in software Developers' workflow and contribution to Sustainability

Theme 1: Benefits of AI and Efficiency Boosters:

This theme elaborates on how AI tools benefit software developers and boost the efficiency of the software developers in their workflows. P7 believes AI tools faster development by reducing the number of lines of code and he said:

'Let's say we had to write like ten lines of code because we didn't know the advanced features of the language, right? Let's say we have to find out on the Internet. But now if we want to generate something advanced, we should give some prompt LLM, and it could go down to like two or three lines, and it still does the same thing. So, I would say it's really good in that area'.

P6 describes how AI tools such as Github Copilot improve productivity through auto code suggestion and generation. Their Company sponsors GitHub Copilot and his observation describes

'So, from our company perspective, they basically sponsor GitHub Copilot. And we have some observations about this because the company wants to improve our productivity and GitHub Copilot is actually a code generation and auto code completion tool that helps to improve our capacity'.

Like other participants, P1 also AI increases software developers through auto code generations and suggestions. AI coding assistant tools also assist developers by providing auto reviews, code refinement and completion of code, innovative suggestions, code documentation, and accurate code. P11 shared his experience like this:

'AI tools in Scrum projects improve technical quality by automating code reviews, identifying performance bottlenecks, and suggesting refactoring opportunities'.

AI tools are good at enhancing the testing process and debugging. AI tools can generate test cases, error detection, error prevention, and fault correction and can perform unit tests. According to P8: *'Yeah, there is this one big scope in testing because it can generate test cases and stuff'.*

P1 believes that AI tools automate repetitive tasks and AI tools reduce human effort by refraining developers from searching in Google, Stack Overflow or any other search engine.

Theme 2: AI as an Augmentative Tool for Routine Coding

This theme aims to polish the routine tasks of software developers. P11 finds out AI tools automate repetitive tasks AI tools also optimize the tasks and perform code refactoring to reduce the repetitive task. AI tools assist developers by writing the full user story to prompt some tasks. P7 detects AI tools that can automate the testing. According to P7: *'It can automate the testing and save time and development cost by detecting errors also'.*

Generative AI such as ChatGPT aids novice developers by providing learning content to contribute to the workflows and sustain the job. AI tools assist developers in learning the coding style of the project so that novice developers don't need to get help from others.

Theme 3: Sustainable Development Through Efficiency and Automation

This theme points to sustainable development through efficiency and automation by reducing development time, personal growth, and Minimized Costs. Software developers notice AI tools such as GitHub Copilot, and ChatGPT provide opportunities to develop efficient maintainable code and optimize the code. P1 observes that Github Copilot can generate efficient and maintainable code, and this tool assists in reducing the usage of resources by optimizing the code. Thus, AI tools reduce the development time. According to P10: *'ChatGPT or GitHub Copilot saves you time by giving you the quotes, suggestions or refactoring the code'.*

AI tools can play a role as a guide for software developers and contribute to their personal growth and development. AI tools are better at providing optimization tips and tricks. P2 Illustrates: *'For my case, what I do is I improve my code quality by the help of AI. Sometimes I write a piece of code and then I want to get more explanation or more, more optimization tricks and tips from the AI'*. AI tools can generate innovative ideas, and P3 thinks ChatGPT can generate constructive ideas that help the software developers to be innovative in the development premises. Nowadays software developers depend on AI tools because tools simplify tasks and make the testing process easier. P2 relies on AI tools for testing purposes for 50% to 60% of cases. P10 depends on ChatGPT because it reduces development time by providing solutions in a quick manner instead of searching in different web browsers.

AI-driven testing processes automate repetitive tasks and help developers to identify bugs faster in an efficient manner and it reduces the development cost. P1 believes that reusable code reduces the development cost and has a good impact on sustainable development. P4 selects the AI tools for faster error detection and error prevention, reduces the development time and cost and contributes to sustainability.

Theme 4: Optimized Workflow and Long-term Code Quality

This theme discusses how AI tools aid software developers in contributing to sustainability by optimizing workflows and maintaining long-term code quality in the software development premises by reducing redundancy in development, code quality and

maintainability, and ongoing code reliability. AI tools contribute to software development workflows by reducing repetitive tasks and shorter coding processes. P4 said: *'I'm going to need like five days or one week to write. So maybe that's why our lives are really getting really easier with the AI tools. Yeah, it's really impressive and yeah, you know, like short development times, so it makes your life easier to that extent'.*

Participants also believe that AI tools like GitHub Copilot, and ChatGPT assist developers to maintain the code quality by providing clean code, detecting patterns, and efficient code. Efficient coding processes always consume fewer resources to develop software. P6 points out that: *'Yes, to software development, sustainability means to me is to write efficient code which actually use, I mean fewer resources like any hardware, fewer resources and it actually consumes more, less resource, less power, everything'*. Software developers use reusable code to make the code scalable and maintainable to reduce the development cost and it is quite economical in terms of the stability of the code. AI tools help to reduce the ineffective, repetitive code to improve the sustainability throughout the project.

5 DISCUSSION

This chapter to dedicated to revealing the findings from this research, it offers an in-depth interpretation of the context of the research questions and related literature. This chapter elaborates on the key findings of the research, compares them with existing research, and discovers unexpected findings and the implications of the studies in both practical and existing research. In this chapter, this study also illustrates the obstacles of the study and anchors on upcoming research work possibilities.

5.1 Restating Research Aim and RQs

The findings of this research are established on a Systematic Literature Review (SLR) and a semi-structured interview process to answer the 2 research questions. The first research question- **What are the key factors, barriers, and benefits influencing software developers' adoption of AI tools in their workflow?** To find out the answer to this research question, a Systematic Literature Review is followed throughout the study. This research question aims to point out the benefits and barriers to software developers' adoption of AI tools in the software creation workflow. It reveals how AI tools benefit software developers by assisting them in the software development process and boosting their efficiency to inspire them to adopt AI tools. AI tools also help developers to perform routine tasks. This research question discovers how these AI tools hinder the software creation process; it marks the barriers to using AI tools in software creation.

The 2nd research question- **What are software developers' thoughts on the advantages of AI technologies in terms of productivity and sustainability?** To reveal the answer to this question, semi semi-structured interview process is organized with 11 software developers around the globe. The interviewees are working in a reputed software company in a scrum team. A set of questions is prepared to get real-time insights from the software developers to find out the influential factors of adopting AI tools such as GitHub Copilot, ChatGPT, Stack Spot AI, Rakuten AI etc in their workflows and contributions to sustainability. This section explores software developers' perceptions of how these AI tools increase their productivity and boost their efficiency practically. In this section, interviewee also expresses their opinion on the AI tools' ability to help in routine coding tasks and enhance their knowledge. In this section, this research question also points out the AI tools' assistance in software development through automation, efficiency, optimize the workflow to contribute to sustainability by reducing development time, and cost, reducing redundancy and maintaining long-term code code quality.

Finally, after analysing these 2 research questions, this study aims to provide a solid idea, of how software developers are using AI tools in their working environments to mitigate their workloads and improve their productivity to develop the best quality software products after

having barriers and their contributions towards sustainability and also highlight the common findings derived from SLR and semi-structured interviews in both Figure 6 and Figure 7.

5.2 Key Findings

Studying the 2 research questions and analysing the data from these research questions, this research discloses the validated data, and these data reveal the Impact of AI tools on productivity and sustainability in the creation of software. Figure 8 presents the data on the Impact of AI tools on Productivity and Sustainability as a theme in software development.

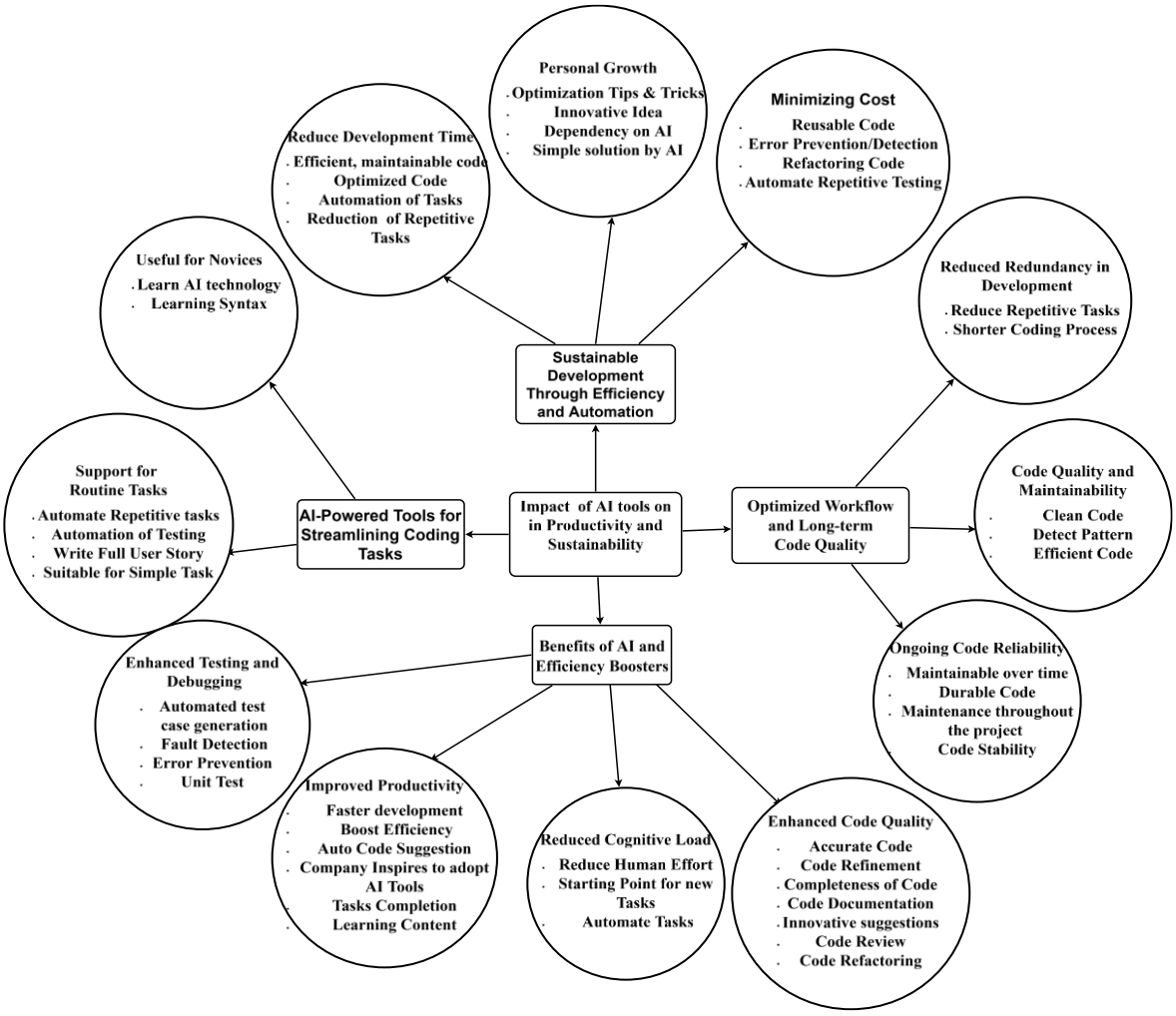

Figure 8: Impact of AI Tools on Productivity and Sustainability

The data suggests that AI tools assist software developers in increasing their productivity and boosting their efficiency by enhancing code quality, reducing cognitive load and enhancing the testing and debugging process. The analysis identifies that auto code suggestions, auto task completion, code refactoring, and innovative suggestions speed the development process, increase productivity and boost the efficiency of the developers. The data also suggest that AI tools automate the testing process by creating test cases, detecting errors and preventing errors to boost the productivity of the software developers. The data supports that AI tools are also good for starting a new task and reducing human efforts. This analysis discloses that AI tools support coding as routing tasks by automating repetitive tasks, automating the testing procedure, and performing simple tasks also presents that AI tools provide opportunities to developers as good sources of learning content.

The analysis identifies that AI tools assist developers in contributing to sustainability through efficiency and automation in tasks. The data suggests AI tools reduce the development time by generating efficient and maintainable code, AI tools also assist in optimizing code and reduce repetitive tasks. The data supports that AI tools reduce the software development cost by detecting error/preventing error and the data also points out that AI tools can refactor the code and reuse the code to minimize the development cost. This data also discloses that AI tools are good for the personal growth of software developers. These AI tools provide innovative ideas to the software developers, optimization tips and tricks to work faster in the development premises. The data suggest that software developers depend on AI tools in their regular activities. The data points out that AI tools optimize workflows and secure long-term code quality to maintain code quality. The data presents AI tools assist to in shorter the coding process, reducing the repetitive tasks to reduce the redundancy in the software development. The data suggests AI tools make code reliable by maintaining it over time, making durable code.

5.2.1 Comparing with Existing Literature and Real-life Practice

This research finds out that software developers are adopting AI tools in their workflows to increase their productivity, and efficiency and improve their code quality that aligns with the existing research. This research points out AI tools improve productivity and speed up the

development process by auto code generations, code completions code refactoring, automate tasks, innovative suggestions automate testing. Prior studies also indicate that AI tools like ChatGPT, and GitHub Copilot faster the software creation process by automating repetitive tasks (Peras et al., 2023), auto code generations (Waseem et al., 2024), and code completion (N. Nguyen & S. Nadi, 2022) and facilitating real-time debugging. This literature review of this research reveals the evidence from research from (Wendy et al., 2024) and (Waseem at el., 2024) and highlights the factors that increase the productivity of software developers while integrating AI tools in the software creation process. In real time developments, the participants also provide their option that AI tools improve the efficiency and productivity through auto code generation, auto code generation and automate repetitive tasks (P6, P7). This research agrees with real-time software developers that AI tools are suitable for simple tasks and provide innovative ideas to start a new tasks, however, (N. Nguyen & S. Nadi, 2022) is concerned that AI tools have the prospective to increase code quality and points out that AI tools boost efficiency, however, to resolve complex tasks AI tools require human intervention to maintain the accuracy of the code. In real development premises, the software developer reveals, AI tools can suggestion ideas to start a new tasks and good option to perform simple tasks like unit testing (P3). This research and (Gustavo et al., 2024), (Purwoko et al., 2023) and participants, all point out that AI tools enhance the code quality, however, AI tools generate inconsistent responses and insecure code in certain circumstances such as complex tasks. This research conveys the same opinion as (S. Mehmood et al., 2023) that AI tools (i.e. GitHub Copilot) automate the testing process and generate test cases to increase productivity. There is not enough research in the field of software development where it indicates software developers contributes to sustainability by adopting AI tools. This research and (Duan et al., 2019) discovered that AI tools assist software developers in automating tasks, generating efficient code, and fixing bugs to reduce operational time and cost and thus contribute to economic sustainability. In real-time software development premises, the software developers and this research point out that AI tools assist to optimize code (P11, P1), reduce repetitive tasks, automate testing process (P3), refactoring code (P10) assist to generate reusable code to reduce the development time and cost to achieve the project milestones, overall to contribute to the economic sustainability. Interviewee participants also disclose, though AI tools have limitation but their dependency

on AI tools is growing because generative AI's provide good source of learning for novice and reduce cognitive load (P4). The participants found out AI tools can generate clean code, and efficient code (P6, P9) and can be maintained for long term. Optimization of code, reduce redundancy in the code and shorter coding process and generate stable code for long period activity.

5.2.2 Explain Unexpected Findings

In this research, a number of unexpected findings are revealed related to the limitations of the AI tools in software development in managing complex development or refined tasks without the participation of humans. In opposition to expectation, AI tools like GitHub Copilot and Stack Spot AI can handle complex coding tasks, the finding points out that the AI tools need human supervision for optimal performance, especially in managing complicated tasks in the software creation process. This is unpredicted that as the AI tools have generative capabilities, it is expected that AI tools fully automate complicated tasks. However, the software developers find out that these AI tools generate generic code that hampers the development workflows rather than a continuous automation process (Gustavo et al., 2024) and these codes need to be refined and as a result, it increases the development time and cost. Additionally, AI tools significantly provide support the novice software developers, however, the research points out that novice developers are struggling to interpret the AI tools' generated solution in their working environment.

5.2.3 Discussion of Implication of Findings

This work of study deals with the fact of adopting AI tools in the development of software by developers working in Scrum. The findings from this research focus on the impact of AI tools on productivity and sustainability. AI tools is dedicated to increasing the productivity of software developers through auto code suggestions, code refactoring, code review and automated testing processes. Thus, AI tools speed up the development process. Besides, AI tools automate repetitive tasks, reduce human efforts and boost the efficiency of code completion, code refinement, and innovative suggestions. Additionally, this study also

points out that AI tools extend an efficient hand to software developers, to make sure sustainability. By optimizing the code, automating the repetitive tasks and generating maintainable code, reusable code AI tools reduce the development time and development cost AI tools speed up the sprint process in Scrum. Thus, by reducing the development cost, AI tools can create a positive impact on the economic dimension of sustainability. Furthermore, AI tools contribute to sustainability by also reducing repetitive tasks. Because it requires less energy consumption and resource waste. This study also reveals that AI tools help to maintain the code quality for the long term. AI tools help to generate efficient code, detect patterns of the code, and clean code. As a result, AI tools help developers to contribute sustainability by reducing development time and cost. In security issues, the study figures out the developers oversee and validate the generated code to prevent vulnerabilities, and risks in the development process and it gives importance to the proper usage of AI tools to maintain the integrity and security in the development premises.

5.3 Limitations of the Study

This section presents the critical limitations of the potential that may have affected the research's outcomes, scope and overall relevance. Although this research sheds light on the usage of AI tools, in software development with a view to increasing software developers efficiency and augmenting sustainability by making sure its utmost efficacy, however, it is very important to recognize the borderline of the research.

The participants' sample size is 11 and they are from different locations and backgrounds, the sample size is limited, and it does not capture all the perspectives of the industries' practices. This study captures the most popular AI tools adopted by software developers in their software development workflows and facilitated by these tools. However technology is evolving every day, and the recently developed AI tools' potential several features may have been excluded. This study is limited to Scrum methodologies only and it is not applied to other methodologies like Kanban, DevOps, and Waterfall.

5.4 Future Works

This study has just begun the journey in the premise of software development, especially software development using AI tools. This study suggests several directions for the future research. Future studies may reveal the impingement of AI tools upon software developers who put all their efforts other software development methodologies such as Kanban, DevOps, Waterfall etc in terms of productivity and sustainability. Future research may point out how AI tools contribute to energy efficiency and resource-management sustainable coding practices. Future research could explore the performance of AI tools across various types of projects and coding environments. Specifically, studies could investigate how well tools like ChatGPT and GitHub Copilot handle complex, domain-specific tasks compared to general or routine coding tasks. The security implications of using AI in coding environments require further exploration. As identified, AI-generated code can be inconsistent and pose security vulnerabilities. Future studies could investigate the frequency and types of security issues introduced by AI tools, developing frameworks for evaluating and improving the security of AI-generated code.

6 CONCLUSIONS

This research scrutinizes the solid idea, of how software developers are using AI tools in their software augmentation workaround to improve their productivity to develop the best quality software and their contributions towards sustainability, overall contributing to the software development process. To get a solid base, this study points out the key factors, benefits, and barriers of adopting AI tools from the existing literature and conducts semi-structured interviews to get the insights of the software developer over the advantages of adopting AI tools in their software creation process (i.e. Scrum) and their contributions to sustainability. This research also points out the constraints of the study and suggests worthwhile scopes for further future research works.

This study concentrates in investigating the impingement of AI tools in the software development workarounds and how AI tools extend an efficient hand in order to augment productivity and sustainability. The results indicate that AI tools (i.e. GitHub Copilot, ChatGPT) act a inevitable role in improving the productivity of software developers by automating repetitive tasks, enhancing code quality, innovative suggestions, and automation of the testing process. Furthermore, the findings highlight that AI tools reduce the cognitive load of the software experts and speed up the software creation process. Additionally, AI tools aid software developers in contributing to sustainability by reducing software development time and cost**.** The findings disclose AI tools assist in optimizing tasks, error detection and prevention, refactoring code and reusable code to reduce the software development time and cost and contribute to economic sustainability. Besides, the findings show that AI tools also assist developers in maintaining code quality and optimising the workflows by guiding and generating clean and stable code and detecting patterns. However, the study also finds barriers to adopting AI tools in handling complex tasks, inconsistent outputs, and security vulnerabilities. The findings present that AI tools aid software developers in increasing their personal knowledge of using AI technology in software development.

This research brings about quite a few impingements to the premise of software development by pointing out key gaps which were prevailing in the existing literature review and offering fresh perspectives on the adoption of AI tools by software developers. This study provides a systematic exploration of the benefits, barriers, and significant influencing factors to inspire the adoption of AI tools by software developers. It reveals the significant benefits of using AI tools such as increased efficiency, enhanced code quality, automated testing process, and reduced cognitive load, while also highlighting barriers, including security vulnerabilities, inconsistent outputs, and the need for human oversight in complex scenarios. This study also captures the firsthand experiences and insights of developers who use AI tools in real-world projects through semi-structured interviews. This qualitative approach offers an in-depth realization of the underlying practical matters of AI tool adoption, including their impact on productivity, skill enhancement and their contributions to sustainability. This study explores how AI tools aid developers in promoting sustainability in the software development process by reducing development costs, time, and resource consumption by optimizing tasks, and

aiding in maintaining clean code, stable code, and automation of tasks. The study lays the groundwork for future research by identifying unexplored areas such as how AI tools contribute to energy efficiency and resource-management sustainable coding practices, the usability of these tools across diverse coding environments, and their contributions to environmental sustainability.

In conclusion, the research emphasises the outstanding role of AI tools in software development premises in productivity and sustainability. Though it has barriers, AI tools significantly assist software developers in continuous innovation, enhance development skills and inspire sustainable development practices to bring long-term revolution in the software industry. This study serves as a foundation for future exploration into leveraging AI to meet the evolving needs of the software industry.

References

G. Pinto, C. de Souza, T. Rocha, I. Steinmacher, A. de Souza and E. Monteiro, "Developer Experiences with a Contextualized AI Coding Assistant: Usability, Expectations, and Outcomes," 2024 IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI (CAIN), Lisbon, Portugal, 2024, pp. 81-91.

Nathalia Nascimento, et al. “COMPARING SOFTWARE DEVELOPERS WITH CHATGPT: AN EMPIRICAL INVESTIGATION.” Artificial Intelligence Vs. Software Engineers: An Empirical Study on Performance and Efficiency Using ChatGPT, 2023, pp. 24–33. ACM DL, dl.acm.org/doi/abs/10.5555/3615924.3615927.

Peras, D., Stapić, Z. & Matijević, M. 2023, AI techniques and tools in Agile Software Development: Preliminary research, Faculty of Organization and Informatics Varazdin, Varazdin.

Waseem M., Das T., Ahmad A., Liang P., Fahmideh M., Mikkonen T. ChatGPT as a Software Development Bot: A Project-Based Study (2024) International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE - Proceedings, pp. 406 - 413

Daniel Fried, Armen Aghajanyan, Jessy Lin, Sida Wang, Eric Wallace, Freda Shi, Ruiqi Zhong, Wen tau Yih, Luke Zettlemoyer, and Mike Lewis. 2023. InCoder: A Generative Model for Code Infilling and Synthesis. arXiv:2204.05999 [cs.SE]

Wendy Mendes, Samara Souza, and Cleidson De Souza. 2024. "You're on a bicycle with a little motor": Benefits and Challenges of Using AI Code Assistants. In Proceedings of the 2024 IEEE/ACM 17th International Conference on Cooperative and Human Aspects of Software Engineering (CHASE '24). Association for Computing Machinery, New York, NY, USA, 144–152. https://doi.org/10.1145/3641822.3641882

V. T. Faniran, A. Badru and N. Ajayi, "Adopting Scrum as an Agile approach in distributed software development: A review of literature," 2017 1st International Conference on Next

Generation Computing Applications (NextComp), Mauritius, 2017, pp. 36-40, doi: 10.1109/NEXTCOMP.2017.8016173.

R. Mdallal, D. Alsawalmeh, D. Jarad and A. Odeh, "AI-Powered Conceptual Model for Scrum Framework," 2023 3rd International Conference on Computing and Information Technology (ICCIT), Tabuk, Saudi Arabia, 2023, pp. 597-604, Doi: 10.1109/ICCIT58132.2023.10273881.

B. Ayoola, "Evaluation of Stakeholder Mapping and Personas for Sustainable Software Development," 2023 IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), Melbourne, Australia, 2023, pp. 270-272, doi: 10.1109/ICSE-Companion58688.2023.00073.

P. Bambazek, I. Groher and N. Seyff, "Application of the Sustainability Awareness Framework in Agile Software Development," 2023 IEEE 31st International Requirements Engineering Conference (RE), Hannover, Germany, 2023, pp. 264-274, doi: 10.1109/RE57278.2023.00034.

Klemmer, Jan H., et al. “Using AI Assistants in Software Development: A Qualitative Study on Security Practices and Concerns.” Using AI Assistants in Software Development: A Qualitative Study on Security Practices and Concerns, simg.baai.ac.cn/paperfile/3dd610f4-ecab-4e3c-975a-2383cd738843.pdf

Roland, V. H. (2021). The impact of AI-based tools on software development work. LUP Student Papers. https://lup.lub.lu.se/student-papers/search/publication/9069046

K. Bhandari, K. Kumar and A. L. Sangal, "Artificial Intelligence in Software Engineering: Perspectives and Challenges," 2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC), Jalandhar, India, 2023, pp. 133-137, doi: 10.1109/ICSCCC58608.2023.10176436.

F. Şahin, "Artificial Intelligence, Game Theory, Programming Used Languages and Platforms, Game Types and Training Methods," 2022 International Conference on Artificial Intelligence of Things (ICAIoT), Istanbul, Turkey, 2022, pp. 1-7, doi: 10.1109/ICAIoT57170.2022.10121899.

D. Sobania, M. Briesch, C. Hanna and J. Petke, "An Analysis of the Automatic Bug Fixing Performance of ChatGPT," 2023 IEEE/ACM International Workshop on Automated Program Repair (APR), Melbourne, Australia, 2023, pp. 23-30, doi: 10.1109/APR59189.2023.00012.

S. Imai, "Is GitHub Copilot a Substitute for Human Pair-programming? An Empirical Study," 2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), Pittsburgh, PA, USA, 2022, pp. 319-321, doi: 10.1145/3510454.3522684.

Tian, Haoye, Weiqi Lu, Tsz On Li, Xunzhu Tang, Shing-Chi Cheung, Jacques Klein, and Tegawendé F. Bissyandé. "Is ChatGPT the ultimate programming assistant--how far is it?." arXiv preprint arXiv:2304.11938 (2023).

J. W. Purwoko, T. Abdullah, B. Wijaya, A. A. Santoso Gunawan and K. E. Saputra, "Analysis ChatGPT Potential: Transforming Software Development with AI Chat Bots," 2023 International Conference on Networking, Electrical Engineering, Computer Science, and Technology (IConNECT), Bandar Lampung, Indonesia, 2023, pp. 36-41, doi: 10.1109/IConNECT56593.2023.10327087.

N. Nguyen and S. Nadi, "An Empirical Evaluation of GitHub Copilot's Code Suggestions," 2022 IEEE/ACM 19th International Conference on Mining Software Repositories (MSR), Pittsburgh, PA, USA, 2022, pp. 1-5, doi: 10.1145/3524842.3528470.

Rudolf Ramler, Michael Moser, Lukas Fischer, Markus Nissl, and René Heinzl. 2024. Industrial Experience Report on AI-Assisted Coding in Professional Software Development. In 2024 International Workshop on Large Language Models for Code (LLM4Code ’24), April 20, 2024, Lisbon, Portugal. ACM, NewYork, NY, USA, 7 pages. https://doi.org/10.1145/3643795.3648377

Gustavo Pinto, Cleidson R. B. de Souza, João Batista Neto, Alberto de Souza, Tarcísio Gotto, and Edward Monteiro. 2024. Lessons from Building StackSpot AI: A Contextualized AI Coding Assistant. In 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP ’24), April 14–20, 2024, Lisbon, Portugal. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3639477.3639751

Chao Liu, Xindong Zhang, Hongyu Zhang, Zhiyuan Wan, Zhan Huang, and Meng Yan. 2024. An Empirical Study of Code Search in Intelligent Coding Assistant: Perceptions, Expectations, and Directions. In Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering (FSE Companion '24), July 15–19, 2024, Porto de Galinhas, Brazil. ACM, New York, NY, USA, 11 pages. ttps://doi.org/10.1145/3663529.3663848

Khanh Nghiem, Anh Minh Nguyen, and Nghi Bui. 2024. Envisioning the Next-Generation AI Coding Assistants: Insights & Proposals. In Proceedings of the 1st ACM/IEEE Workshop on Integrated Development Environments (IDE '24). Association for Computing Machinery, New York, NY, USA, 115–117. https://doi-org/10.1145/3643796.3648467

Yan Chen, Sang Won Lee, Yin Xie, YiWei Yang, Walter S. Lasecki, and Steve Oney. 2017. Codeon: On-Demand Software Development Assistance. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17). Association for Computing Machinery, New York, NY, USA, 6220–6231. https://doi.org/10.1145/3025453.3025972

Saklamaeva, V. & Pavlič, L. 2024, "The Potential of AI-Driven Assistants in Scaled Agile Software Development", Applied Sciences, vol. 14, no. 1, pp. 319.

Duan, Y., Edwards, J. S., & Dwivedi, Y. K. (2019). Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda. International Journal of Information Management, 48, 63-71. https://doi.org/10.1016/j.ijinfomgt.2019.01.021

Barenkamp, M., Rebstadt, J., & Thomas, O. (2020). Applications of AI in classical software engineering. AI Perspectives, 2(1). https://doi.org/10.1186/s42467-020-00005-4

S. Shafiq, A. Mashkoor, C. Mayr-Dorn and A. Egyed, "A Literature Review of Using Machine Learning in Software Development Life Cycle Stages," in IEEE Access, vol. 9, pp. 140896-140920, 2021, doi: 10.1109/

E. Trinh, M. Funke, P. Lago and J. Bogner, "Sustainability Integration of Artificial Intelligence into the Software Development Life Cycle," 2024 IEEE 21st International Conference on Software Architecture Companion (ICSA-C), Hyderabad, India, 2024, pp. 199-206, doi: 10.1109/ICSA-C63560.2024.00044.

H. Hourani, A. Hammad and M. Lafi, "The Impact of Artificial Intelligence on Software Testing," 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman, Jordan, 2019, pp. 565-570, doi: 10.1109/JEEIT.2019.8717439.

G. Pinto, C. R. B. de Souza, J. B. Neto, A. de Souza, T. Gotto and E. Monteiro, "Lessons from Building StackSpot Al: A Contextualized AI Coding Assistant," 2024 IEEE/ACM 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Lisbon, Portugal, 2024, pp. 408-417, doi: 10.1145/3639477.3639751

Steven I. Ross, Fernando Martinez, Stephanie Houde, Michael Muller, and Justin D. Weisz. 2023. The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development. In Proceedings of the 28th International Conference on Intelligent User Interfaces (IUI '23). Association for Computing Machinery, New York, NY, USA, 491–514. https://doi.org/10.1145/3581641.3584037

N. Davila et al., "An Industry Case Study on Adoption of AI-based Programming Assistants," 2024 IEEE/ACM 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), Lisbon, Portugal, 2024, pp. 92-102, doi: 10.1145/3639477.3643648.

Asha Rajbhoj, Akanksha Somase, Piyush Kulkarni, and Vinay Kulkarni. 2024. Accelerating Software Development Using Generative AI: ChatGPT Case Study. In Proceedings of the 17th Innovations in Software Engineering Conference (ISEC '24). Association for Computing Machinery, New York, NY, USA, Article 5, 1–11. https://doi-10.1145/3641399.3641403

Saklamaeva, V. & Pavlič, L. 2024, "The Potential of AI-Driven Assistants in Scaled Agile Software Development", Applied Sciences, vol. 14, no. 1, pp. 319.

Peng, S., Kalliamvakou, E., Cihon, P. and Demirer, M., 2023. The impact of ai on developer productivity: Evidence from GitHub copilot. arXiv preprint arXiv:2302.06590.

S. Mohamed, A. Parvin and E. Parra, "Chatting with AI: Deciphering Developer Conversations with ChatGPT," 2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR), Lisbon, Portugal, 2024, pp. 187-191.

S. Mehmood, U. I. Janjua and A. Ahmed, "From Manual to Automatic: The Evolution of Test Case Generation Methods and the Role of GitHub Copilot," 2023 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 2023, pp. 13-18, doi: 10.1109/FIT60620.2023.00013.

D. Sobania, M. Briesch, C. Hanna and J. Petke, "An Analysis of the Automatic Bug Fixing Performance of ChatGPT," 2023 IEEE/ACM International Workshop on Automated Program Repair (APR), Melbourne, Australia, 2023, pp. 23-30, doi: 10.1109/APR59189.2023.00012.

Daniel Guaman, Pablo Alejandro, Paola Cabrera"SonarQube as a Tool to Identify Software Metrics and Technical Debt in the Source Code through Static Analysis." Proceedings of 2017 the 7th International Workshop on Computer Science and Engineering, Jan. 2017, https://doi.org/10.18178/wcse.2017.06.030.

J. W. Purwoko, T. Abdullah, B. Wijaya, A. A. Santoso Gunawan and K. E. Saputra, "Analysis ChatGPT Potential: Transforming Software Development with AI Chat Bots," 2023 International Conference on Networking, Electrical Engineering, Computer Science, and Technology (IConNECT), Bandar Lampung, Indonesia, 2023, pp. 36-41, doi: 10.1109/IConNECT56593.2023.1032708

D. -K. Kim, J. Chen, H. Ming and L. Lu, "Assessment of ChatGPT's Proficiency in Software Development," 2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE), Las Vegas, NV, USA, 2023, pp. 2637-2644, doi: 10.1109/CSCE60160.2023.00421.

M. Kanbur, O. P. C and P. Kulkarni, "Creative AI in Software Project Management," 2023 2nd International Conference on Futuristic Technologies (INCOFT), Belagavi, Karnataka, India, 2023, pp. 1-9, doi: 10.1109/INCOFT60753.2023.10425234

S. Sikand, R. Mehra, V. S. Sharma, V. Kaulgud, S. Podder and A. P. Burden, "Do Generative AI Tools Ensure Green Code? An Investigative Study," 2024 IEEE/ACM International Workshop on Responsible AI Engineering (RAIE), Lisbon, Portugal, 2024, pp. 52-55

Freitag, Charlotte & Berners-Lee, Mike & Widdicks, Kelly & Knowles, Bran & Blair, Gordon & Friday, Adrian. (2021). The real climate and transformative impact of ICT: A

critique of estimates, trends, and regulations. Patterns. 2. 100340. 10.1016/j.patter.2021.100340.

Gustavo Pinto and Fernando Castor. 2017. Energy efficiency: a new concern for application software developers. Commun. ACM 60, 12 (December 2017), 68–75. https://doi.org/10.1145/3154384

R. Khoury, A. R. Avila, J. Brunelle and B. M. Camara, "How Secure is Code Generated by ChatGPT?" 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Honolulu, Oahu, HI, USA, 2023, pp. 2445-2451, doi: 10.1109/SMC53992.2023.10394237.

P. Bambazek, I. Groher and N. Seyff, "Sustainability in Agile Software Development: A Survey Study among Practitioners," 2022 International Conference on ICT for Sustainability (ICT4S), Plovdiv, Bulgaria, 2022, pp. 13-23, doi: 10.1109/ICT4S55073.2022.00013.

Angela Carrera-Rivera, William Ochoa, Felix Larrinaga, Ganix Lasa, How-to conduct a systematic literature review: A quick guide for computer science research,MethodsX,Volume9,2022,101895,ISSN2215161,https://doi.org/10.1016/j.mex.2022.101895.

Johnson, R Burke, Anthony J Onwuegbuzie, and Lisa A Turner (2007). “Toward a definition of mixed methods research”. In: Journal of mixed methods research 1.2, pp. 112–133

Boell, S.K. & Cecez-kecmanovic, D. 2015, "On being 'systematic' in literature reviews in IS", Journal of Information Technology, suppl.Special Issue: Revisiting Strategic Alignment 25 Years On, vol. 30, no. 2, pp. 161-173.

Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., & Group, a.t. P. (2009). Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement. Annals of Internal Medicine, 151(4), 264–269. https://doi.org/10.7326/0003-4819-151-4-200908180-00135. %J Annals of Internal Medicine

S. Jalali and C. Wohlin, "Systematic literature studies: Database searches vs. backward snowballing," Proceedings of the 2012 ACM-IEEE International Symposium on Empirical

Software Engineering and Measurement, Lund, Sweden, 2012, pp. 29-38, doi: 10.1145/2372251.2372257

S. E. Hove and B. Anda, "Experiences from conducting semi-structured interviews in empirical software engineering research," 11th IEEE International Software Metrics Symposium (METRICS'05), Como, Italy, 2005, pp. 10 pp.-23, doi: 10.1109/METRICS.2005.24.

Cohen, L., Manion, L., & Morrison, K. (2017). Research Methods in Education (8th ed.). Routledge. https://doi.org/10.4324/9781315456539

Vaismoradi, M., Turunen, H., & Bondas, T. (2016). Theme development in qualitative content analysis and thematic analysis. Journal of Nursing Education and Practice, 3(1), 1-10. https://doi.org/10.5430/jnep.v3n1p1.

A APPENDIX: SOURCE OF SLR KEY FINDINGS

Type	**References**
Benefits	(Waseem et al., 2024), (Gustavo et al., 2024), (Nathalia et al., 2023), (Asha et al., 2024), (Wendy et al., 2024), (S. Mohamed et al, 2024), (Peras et al., 2023), (D. Sobania et al.,2023), (N. Nguyen & S. Nadi ,2022)
Barriers	(Waseem et al., 2024), (Wendy et al., 2024), Gustavo et al., 2024), Asha et al., 2024), (J. W. Purwoko et al., 2023), (S. Imai ,2022)

B APPENDIX: INTERVIEW QUESTIONS

Personal Background:

- Your name:
- Name of your Company:
- Your role in your Company:
- No. of years Working in the Software Industry:

- No. of years working on Software Development Projects:

- No. of years working on Agile Projects:

- What types of projects have you worked on (e.g., web applications, mobile apps, enterprise software)

General Thoughts on AI Technologies:

- What is your workplace's main software development methodology (e.g., Scrum, Kanban, DevOps. Waterfall)? What are the primary motivations or key points for adopting AI tools in software development?

Related to AI tools:

- Which AI tools are you using in the Software development process in programming, code review, and program repair? (i.e Tabnine, Intellicode, GitHub copilot, DeepCode, Code Guru, RewardRepair, CURE or ChatGPT etc)

- Are these AI tools recommended by your company? If it is recommended, how was your team's response compared to traditional tools?

- Can you explain some limitations of these AI tools in comparison to traditional tools in the software development process?

- Do you use any AI tools in Scrum to report sprints? (i.e. Stepsize AI, Github Insights, Jira with Jira Align etc) ?

Advantages of AI technologies in terms of Productivity:

- As a professional software developer, how do you compare your professional life in software development with traditional tools vs AI tools vs Without AI tools (stop using AI tools)

- What are the main advantages you've experienced using these AI tools in your work over traditional tools (i.e. code generation/ optimization, debugging, error detection)?

Sustainability:

- What does sustainability in software development mean to you? How do these AI tools can contribute to it?

- In your experience, how have these AI tools helped improve the technical quality of the software (e.g., code quality, performance) in Scrum projects?

- How do AI tools contribute to sustainability in the testing and debugging phases? What's your opinion?

- Can you share an instance where AI tools reduced resource consumption, development time, and cost sustainably?

- As a software developer, do you think these AI tools promote sustainability better than the traditional way in the scrum process?