Investigating the Key Factors Influencing Software Developer Productivity in Developing Regions

By

Rafid Al Ahsan

Student ID: **2321678**

Summer, 2025

Supervisor:

Mahady Hasan

Associate Professor

Department of Computer Science & Engineering

Independent University, Bangladesh

7th September,2025

Dissertation submitted in partial fulfillment for the degree of Master of Science in Software Engineering

Department of Computer Science & Engineering

Independent University, Bangladesh

Attestation

*** Write about the originality of your work ***

Signature

Date

Name

Acknowledgement

First and foremost, I am deeply grateful to the Almighty Allah for granting me the strength, patience, and determination to complete this thesis. I would like to extend my heartfelt appreciation to my respected supervisor, Mahady Hasan, whose guidance, constructive feedback, and constant support have been invaluable throughout the course of this research. I am also thankful to the faculty members and administrative staff for their academic guidance and logistical assistance during my Master's program. My sincere thanks go to the software developers who participated in the survey; their honest contributions made this study possible. I wish to acknowledge my friends and classmates for their encouragement, support, and companionship during this journey. Finally, I am forever indebted to my parents and family for their unconditional love, prayers, and emotional support, which have been my greatest source of strength and motivation.

Letter of Transmittal

Date: 7th September, 2025

To
Dr. Mahady Hasan
Associate Professor
Independent University, Bangladesh
Bashundhara, Dhaka

Subject:Thesis Report Submission

Dear Sir,

I am pleased to submit my thesis titled "Investigating the Key Factors Influencing Software Developer Productivity in Developing Regions". This report has been prepared for fulfilling the requirement for Master of Science in Software Engineering at Independent University, Bangladesh. This thesis explores the key factors affecting the productivity of software developers in Bangladesh.

I respectfully submit this report for your kind evaluation and approval. I sincerely hope it meets the academic and research standards expected by the department.

Thank you for your time and consideration.

Sincerely,
Rafid Al Ahsan
ID: 2321678

Evaluation Committee

Supervision Panel

.....	.....
Supervisor	Co-supervisor

External Examiners

.....	.....
External Examiner 1	External Examiner 2

Office Use

.....	.....
Program Coordinator	Head of the Department

.....

Director Graduate Studies, Research and Industry Relations

.....

Library

Contents

List of Figures

List of Tables

Abstract

Software developer productivity is a critical factor influencing the efficiency and success of software projects. However, understanding the key factors that impact developer productivity remains a challenge, particularly in region-specific contexts such as Bangladesh. This study aims to identify the primary factors affecting developer productivity and analyze how these factors vary across companies and experience levels. To achieve this, a survey was conducted among software developers working in Bangladesh. The survey was sent out to 168 software developers working across 38 different software firms. The survey yield 44 responses. Statistical analysis was employed to determine the the most influential factors. The findings suggest that autonomy in decision-making, task variety, and using the best practices, software and tool are the strongest factors affecting productivity. While some factors, such as task variety, remain consistent across companies, others, such as remote work policies and communication styles, vary significantly. Additionally, beginner developers prioritize learning and tool adoption, mid-level developers face challenges with tight deadlines, and senior developers focus more on leadership, collaboration and decision-making autonomy.

Keywords— Software Developer Productivity, Productivity Factors, Software Engineering

Chapter 1

Introduction

1.1 Overview

Software development has become an integral part of the global economy, driving innovation and digital transformation across industries. As the demand for software solutions grows, the productivity of software developers has emerged as a critical factor in ensuring project success and organizational efficiency. Bangladesh in recent years had significant IT growth with the number of small software firms rising.

This thesis explores the key factors that influence the productivity of software developers in Bangladesh, with a particular focus on small and medium-sized enterprises. The study seeks to identify the challenges, working conditions, and contextual elements that shape developer performance. Using a survey-based approach, it examines how aspects such as skill level, autonomy, work environment, and access to tools affect developers at different career stages. It also considers how these factors vary across organizations and work arrangements, providing findings that are relevant to the Bangladeshi context while also contributing to the wider discussion on global software development practices.

1.2 Contribution of the thesis

The exiting studies mostly mostly address global context and large corporations. Only a few studies explore the challenges faced by developers in developing regions like Bangladesh where the IT sector is still emerging and software firm are operating with very small team of 10-15 developers. This thesis seeks to bridge that gap by offering an in-depth analysis of the factors influencing developer productivity within Bangladesh's growing software industry.

This study provides valuable insights into how developer productivity is shaped by individual, organizational, and experiential factors. These findings can help companies formulate targeted strategies aimed at enhancing both developer efficiency and job satisfaction. By examining how these factors vary across different career stages, this research delivers tailored insights that organizations can use to support productivity at each level of a developer's professional journey.

1.3 Organization of the thesis

The thesis is structured into six chapters:

- **Chapter 1: Introduction** — Provides an overview of the research, highlights the research gap, states the contributions of the study, and outlines the organization of the thesis.
- **Chapter 2: Literature Review** — Reviews existing studies related to software developer productivity, identifies key findings, and discusses the limitations in prior research that this study seeks to address.
- **Chapter 3: Project Management** — Outlines the project planning and execution phases. This chapter includes the activity list, Work Breakdown Structure (WBS), Gantt Chart, and Network Diagram that guided the timeline and scope of the research project.
- **Chapter 4: Investigating the Key Factors Influencing Software Developer Productivity in Developing Regions** — Presents the main research conducted in this thesis. It includes the problem statement, research methodology, purpose of the study, data collection, research method, data analysis, proposed solution, and result analysis. It also covers demographic results and exploratory analysis.
- **Chapter 5: Sustainability and Ethics** — Discusses the project's sustainability from economic, social, and environmental perspectives, and addresses the ethical considerations involved in the research.
- **Chapter 6: Conclusion and Future Work** — Summarizes the key findings of the study, reflects on the outcomes, and proposes directions for future research.

Each chapter builds upon the previous one, guiding the reader through the full scope of the research—from motivation and planning, through execution and results, to broader impact and future directions.

Chapter 2

Literature Review

2.1 Literature Selection Procedure

The literature selected for this review was chosen based on its relevance to the productivity of software developers. Both qualitative and quantitative studies were considered, with a strong focus on peer-reviewed articles from reputable journals and conferences. Priority was given to survey-based research; however, studies employing empirical data, mixed-method approaches, diary entries, interviews, code-based metrics (such as code lines written), peer assessments, and biometric sensors were also included. In addition, comprehensive reviews were considered that explored various developer environments to ensure a well-rounded understanding of the topic.

2.2 Keywords Searched

Selecting the right keywords is a critical part of conducting an effective and focused literature review. To ensure a comprehensive and relevant search, a wide range of keyword combinations were used across different academic databases. These keywords were carefully chosen to capture both technical and contextual aspects of software developer productivity, including environmental factors, remote work, emotional influences, regional challenges, and modern development tools.To provide clarity and organization, the keywords were grouped into the following thematic categories:

- **General Software Productivity and Performance**
  - Software Engineering Productivity
  - Measuring Developer Productivity
  - Software Developer Performance
  - Measuring Productivity in Software Teams
  - Developer Productivity Metrics
  - Empirical Studies on Software Engineering Practices

- **Contextual and Environmental Factors**
  - Work Environment in Software Development
  - Environmental Factors Affecting Developer Focus and Output
  - Psychological and Emotional Aspects of Developer Productivity
  - Work-from-Home Productivity among Developers
  - Remote Work and Productivity in Software Engineering
- **Technology and Tools**
  - AI Tools for Software Developer Productivity
  - Developer Tooling and Automation Impact on Workflow Efficiency
- **Regional and Socioeconomic Context**
  - Regional Software Development Productivity
  - Developer Challenges in Developing Countries

2.3 Databases for Search

Academic databases such as Google Scholar, IEEE Xplore, ACM Digital Library, Research Gate, SpringerLink, and ScienceDirect were used to find relevant peer-reviewed articles, conference papers and journals.

2.4 Inclusion/Exclusion Criteria

For this study, only research published between 2018 and 2024 was considered, as this timeframe ensures that the findings reflect recent trends and practices. Studies were included if they presented clear empirical evidence, used a sound and well-defined methodology, and were directly related to software engineering productivity.

On the other hand, studies were excluded if they were outdated, lacked peer review, or were based mainly on opinions rather than data. Research with weak or unclear methods was also omitted. In addition, papers that did not show a clear connection to software developer productivity within the context of software engineering were left out. .

2.5 Filtering Results

Out of an initial pool of over 80 papers, filtering was done based on relevance to developer productivity, methodological quality, and citation count. After filtering, 12 core studies were selected for detailed analysis.

2.6 Impact of the project

This literature review lays the foundation for understanding the multifaceted contributors to developer productivity. It justifies the importance of analyzing workplace conditions, emotional well-being, and tool support to improve software development outcomes.

2.7 Literature Review of Papers

Research has consistently shown that the physical work environment and communication structures play a substantial role in shaping software developer productivity [1]. A mixed-method survey involving 1,159 participants demonstrated that factors such as workspace layout, furniture, and general surroundings directly influence performance. In a similar vein, [2] conducted a quantitative study within a major Chinese IT company, revealing that working from home (WFH) can have both beneficial and adverse outcomes, with its effects depending on variables such as project type, team size, programming language, and project maturity [3]. Their analysis of 624 surveys and 2,899 self-reports highlighted that self-assessed productivity often provides a more reliable measure of team productivity, particularly in dynamic settings. While meetings were often deemed counterproductive during development, they were considered useful in planning and release phases. Further evidence from [4], based on responses from 5,971 participants, identified email interruptions and frequent notifications as significant contributors to unproductive days. Similarly, [5], drawing on interviews, surveys, and diary studies, reported that long build times, unstable tests, and frequent meetings were common causes of "bad days" for developers. In addition, [6] found that higher perceptions of code quality often enhance productivity, though the reverse does not necessarily apply. Effective time management also emerged as a critical element, with developers who establish realistic goals and track their progress better able to optimize resources. Supporting studies on goal-setting and persuasive technologies reaffirm the link between structured task management and improved productivity. Beyond technical aspects, emotional states play a key role: [7] demonstrated that emotions can significantly affect the cognitive abilities required for software development. Likewise, [8], based on a survey of 640 participants, concluded that developers who perceive their contributions as meaningful and who work in a positive culture report greater job satisfaction and productivity. Recent technological advances also contribute to this field. Tools such as GitHub Copilot have been shown to increase developer efficiency [9], while pilot studies like [10] suggest that generative AI systems can save time and enhance output. In the context of local industries, evidence suggests that many IT companies fail to integrate security processes throughout the software development life cycle. A study of 15 Bangladeshi firms revealed that, despite awareness of best practices, security measures were rarely implemented effectively [11]. In the area of web development, React has become one of the leading JavaScript libraries. A comparative evaluation of testing tools—Jest, Enzyme, and Cypress—found Enzyme to be the most effective option due to its ease of integration and efficiency [12]. The importance of software architects is also highlighted, as [13] presented empirical methods, grounded in industry practice, to guide

decision-making prior to development. Testing practices are undergoing significant transformation with the adoption of Artificial Intelligence (AI). Studies show that machine learning and deep learning approaches can successfully automate tasks such as test case generation, defect prediction, prioritization, metamorphic testing, Android testing, and white-box validation. Integrating AI into testing has been found to improve efficiency and simplify workflows [14]. Moreover, [15] examined the effects of COVID-19 on timely project delivery across Software Development Life Cycle (SDLC) models. During the pandemic, attentional focus, team stability, communication, maturity, and user involvement emerged as the most critical factors, while before the pandemic, team capability, infrastructure, and commitment were identified as dominant influences. The Software Engineering Body of Knowledge (SWEBOK) also continues to shape the discipline. [16] analyzed existing proposals regarding its content and structure, suggesting revisions to make it more practical and beneficial for innovators, in-house teams, and contractors. The spread of misinformation on social media has attracted research interest, with [17] evaluating four algorithms—Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting—for fake news detection. Results showed that Decision Tree achieved the highest accuracy (99.60%), closely followed by Gradient Boosting (99.55%), Random Forest (99.10%), and Logistic Regression (98.99%). Meanwhile, with the growth of internet use, cyber vulnerabilities are also expanding. [18] conducted an evaluation using BurpSuite, ZAP, and Netsparker to identify recurring web application weaknesses, with results showing the services and transport sectors as most at risk. Parallel advancements in neural networks, data availability, and computing power have also enabled progress in handwriting analysis. [19] used Bangla handwritten characters and deep learning approaches such as CNNs and RNNs to predict writers' gender, achieving an accuracy of 91.85%. Concerns around technology also extend to its social consequences. Studies report that excessive social media usage contributes to addiction, disrupting daily life and negatively impacting mental health. [20], using data from 1,417 participants and ML algorithms including Decision Tree, Random Forest, SVC, k-NN, and Multinomial Naive Bayes, highlighted how platforms like Twitter, Facebook, and Snapchat can significantly contribute to depression and anxiety. Other contributions examine Software Process Improvement (SPI), with [21] proposing a model for applying SPI at CMMI Level 2, based on the experiences of small and medium-sized Bangladeshi firms. Social media marketing has also been studied for its persuasive potential: [22] found that eco-friendly messages on social platforms can shape consumer attitudes, decisions, and sustainable practices. Sentiment analysis remains another growing area. Using the IMDb movie dataset, [23] compared three deep learning models—SNN, CNN, and RNN—finding RNN to achieve the highest accuracy (86.46%). Predictive analytics has also been applied in other areas. For example, [24] applied Random Forest to predict automobile prices based on features like age, mileage, and condition, achieving 99.59% accuracy. In energy studies, [25] proposed a big data–driven solution for forecasting Bangladesh's national energy balance, intended to improve security and sustainability. Similarly, [26] analyzed agile adoption in developing countries, highlighting challenges and offering practical recommendations for Bangladeshi firms.

Continuing in this line, software error detection has also been studied in relation to re-

source constraints such as time, budget, and workforce availability. [27] proposed a system that applies machine learning techniques to analyze system error logs, allowing for real-time error detection through classification algorithms. Their approach demonstrates how ML can strengthen error identification and improve overall system reliability. Similarly, research by [28] applied marketing-mix theory to the Bangladesh software sector, examining the impact of factors such as price, promotion, and after-sales service on dimensions of brand equity. Findings revealed that while some marketing-mix elements correlated strongly with brand equity, others had limited influence. Quality assurance (QA) within the Software Development Life Cycle (SDLC) has also been a focal point of scholarly attention. A review of 140 papers by [29] showed that well-implemented QA practices reduce costs, raise product quality, improve compliance, and boost customer satisfaction, underlining QA's strategic role in the SDLC. Addressing firm-level challenges, [30] argued that software firms in Bangladesh often fall short of growth expectations due to high project diversity. They suggested a decision-making framework to mitigate risks linked to diversification. At a broader level, [31] explored profitability issues in the global software industry, noting that conventional project management often fails when projects require extensive customization. To address this, they proposed a supplementary process model to enhance sustainability and profitability. Research also continues to emphasize client-developer collaboration. For example, [32] introduced a hybrid model combining traditional requirements engineering with agile practices such as user stories, prototypes, and iterative feedback. The approach was found to improve communication and clarity, mitigating common challenges linked to limited client engagement. Governance factors further contribute to project outcomes. A survey of Bangladeshi firms by [33] highlighted that robust project governance frameworks reduce software risks and enhance performance, with project leadership moderating this relationship positively. In the domain of algorithmic research, [34] investigated the problem of furthest k-neighbor queries for continuous datasets. They introduced an efficient grid-based algorithm aimed at minimizing the number of grid cells accessed during computation. Testing on real datasets demonstrated both efficiency and scalability. A related strand of research focuses on process frameworks such as Capability Maturity Model Integration (CMMI). As reviewed by [35], CMMI offers a flexible approach to improving organizational performance, efficiency, and customer satisfaction. However, its complexity demands careful planning and significant resources, though integrating it with Agile or Lean Six Sigma may yield enhanced outcomes. In higher education, [36] examined the integration of QA metrics into a Learning Management System (LMS), emphasizing how comprehensive QA frameworks can ensure fairness, enrich learning, and foster skill development.Startup sustainability has also been studied from a requirements engineering perspective. [37] argued that many startups fail due to neglect of formal requirements processes, and proposed a lightweight, cost-effective framework for adopting formal RE practices to improve long-term survival. Education-focused studies include [38], who examined the use of big data analytics in implementing an Outcome-Based Education (OBE) system in Bangladesh. Their work highlighted how OBE can align learning processes with clearly defined educational outcomes. Complementing this, [39] tested three predictive models—OLS, ANN, and ANFIS—for forecasting Cumulative Grade Point Average

(CGPA). Results confirmed that predictive modeling can provide valuable insights into educational performance trends and demographic factors. Turning to creative software contexts, [40] emphasized the unique complexity of video game development compared to traditional system or application software. Game design was shown to require not only technical completion but also delivery of entertainment value, complicating both development and testing processes. Similarly, [41] focused on environmental applications of deep learning, using neural network models such as VGG-19 and ResNet50 to classify local Bangladeshi wildflowers. Their experiments achieved high accuracy, with VGG-19 yielding 99.2%. In financial technologies, [42] examined challenges faced by commercial banks in Bangladesh during system upgrades, emphasizing the role of local software companies in offering secure and cost-effective digital solutions to sustain competitiveness. In biometric applications, [43] developed a Deep Convolutional Neural Network (D-CNN) to predict blood group and gender using fingerprint data. Their model achieved 99.968% accuracy, showcasing the potential of biometric-based prediction. Environmental sustainability within IT has also gained prominence, with [44] identifying factors such as awareness, knowledge, and perception as key drivers of Green Information Technology (GIT) adoption among Bangladeshi IT professionals. Freelance work trends have also posed challenges for software firms. [45] warned that the shift toward freelancing reduces the availability of skilled professionals, undermining long-term stability, and recommended economic and managerial strategies to address turnover. Software modeling, particularly in Model-Driven Engineering (MDE), has been another area of investigation. [46] explored the barriers facing Bangladesh's software industry in adopting model-based engineering. Their findings point to the need for targeted interventions to promote more comprehensive adoption of system modeling. Productivity research itself has a long tradition in software engineering. [47] synthesized a wide range of studies to compile frequently cited productivity factors, emphasizing that soft or human-centered factors remain underexplored compared to technical ones. These insights provide a foundation for building stronger productivity models in future work. Deep learning (DL) has become central to many recent studies. However, [48] emphasized that while DL shows great promise, building production-ready systems remains difficult for organizations lacking robust infrastructure or large research teams. Drawing on seven industry projects, they identified 12 key challenges spanning development, production, and organizational contexts, stressing that DL, compared to traditional software engineering, is still relatively immature. Broader ICT-related challenges have also been documented. [49] outlined how areas like cloud computing, big data, IoT, and cyber-physical systems require stronger methodological foundations within software engineering. Emerging technologies such as blockchain have also reshaped research priorities. [50] highlighted how blockchain growth necessitates new roles, improved testing strategies, and novel architectural tools. They proposed several research directions to enhance collaboration, testing, and smart contract development. Relatedly, [51] surveyed the use of Large Language Models (LLMs) in software engineering tasks, including coding, design, testing, and documentation. While LLMs bring creativity and versatility, limitations such as hallucinations mean hybrid approaches are often necessary, combining traditional engineering techniques with LLMs for more reliable outcomes. Requirement engineering (RE) practices within agile methods have

been explored extensively. [52] reviewed agile RE challenges, noting that while agile works well in small projects, scaling introduces significant impediments, especially in requirements definition. Similarly, [53] reviewed the integration of generative AI tools like GitHub Copilot and ChatGPT into software development. Their study catalogued 78 research questions across 11 domains, showing that GenAI has the potential to automate parts of software engineering, though its adoption also raises new managerial and ethical challenges. [54] conducted a systematic review of agile RE practices, identifying 17 core practices along with recurring challenges, highlighting areas that require further empirical research, particularly around non-functional requirements and self-organizing teams. Machine learning (ML) systems development has also drawn increasing focus. [55] mapped out a four-stage ML development process consisting of problem understanding, data handling, model building, and monitoring. They identified issues such as unclear client metrics, absence of standardized processes, and database design flaws as major obstacles. To address these, they proposed practical checklists to improve task visibility and support consistent metric identification. Complementing this, [56] reviewed 22 studies on agile RE practices, noting recurring activities and persistent challenges that still limit agile's effectiveness in requirements engineering.

The issue of research methodology itself has also been raised. [57] highlighted widespread weaknesses in empirical software engineering research, particularly regarding sampling strategies. They argued that a lack of representative samples has led to a "generalizability crisis" and recommended more rigorous guidelines while noting that non-probability sampling can still be suitable for qualitative inquiries. Further insights into collaboration processes were provided by [58], who surveyed 749 integrators in pull-based development projects. They found that integrators face difficulties balancing quality assurance with contribution prioritization, underscoring their central decision-making role in open-source ecosystems. Broader process integration issues have also been discussed. [59] argued that software development stages such as planning, testing, integration, and deployment often remain disconnected, undermining efficiency. They introduced the idea of "BizDev," which stresses the continuous alignment of business goals with development activities. Their proposed "Continuous*" roadmap outlines practices for achieving greater reliability and resilience in complex systems. Finally, AI integration in agile settings has been studied by [60], who examined Microsoft teams building AI applications. They noted that AI introduces unique complications, including complex data pipelines, the need for specialized expertise in model customization, and difficulties in modularizing AI components due to interdependencies and unpredictable error behavior. Together, these studies illustrate the breadth of research in software engineering, spanning developer productivity, agile and requirements practices, quality assurance, machine learning, AI integration, and domain-specific challenges such as security, sustainability, and education. Collectively, the literature provides not only insights into existing challenges but also practical frameworks and future directions for strengthening both technical and human aspects of software engineering.

Chapter 3

Project management

3.1 Project Management Overview

To ensure systematic progress and timely completion of the research paper titled "Investigating Factors Affecting Software Developer Productivity in Developing Regions", fundamental project management principles were applied throughout the duration of the thesis. The project was organized into several phases, covering the entire research process from initiation and planning through to final submission.

To manage the work effectively, a Work Breakdown Structure (WBS) was developed using a diagramming tool, which divided the project into smaller, manageable tasks and subtasks. This breakdown helped clarify the deliverables and responsibilities at each stage. A Gantt chart was also prepared using a scripting tool to visualize the timeline, task durations, and dependencies, making it easier to track progress over time. In addition, a network diagram was created with a project management application to highlight task relationships, critical paths, and sequencing.

By combining these tools, the project followed an organized workflow guided by milestones and clear tracking mechanisms. They enabled ongoing monitoring, adjustment of timelines when necessary, and clear documentation of progress. This structured approach significantly contributed to maintaining research focus, avoiding scheduling conflicts, and aligning efforts with the academic calendar and thesis submission deadlines.

3.2 Activity List

The following table outlines all major activities involved in the research project. These activities are derived from the Work Breakdown Structure and form the basis for the Gantt chart and network diagram that follow.

ID	Activity Name	Duration	Predecessor(s)	Remarks
1	Literature Review	16 days	—	Initial exploration
2	Finalize Research Topic	4 days	1	Supervisor-approved
3	Formulate Research Questions	3 days	2	Based on gaps identified
4	Design Methodology & Survey	4 days	3	Includes questionnaire
5	Data Collection	31 days	4	Survey distributed
6	Data Analysis	11 days	5	Use SPSS / Excel
7	Interpret Results	13 days	6	Draw insights
8	Writing Research Paper	8 days	7	All sections
9	Review and Revision	3 days	8	Final proofing
10	Submit Final Paper	1 day	9	Final submission

Table 3.1: Activity List for the Research Project

3.3 WBS

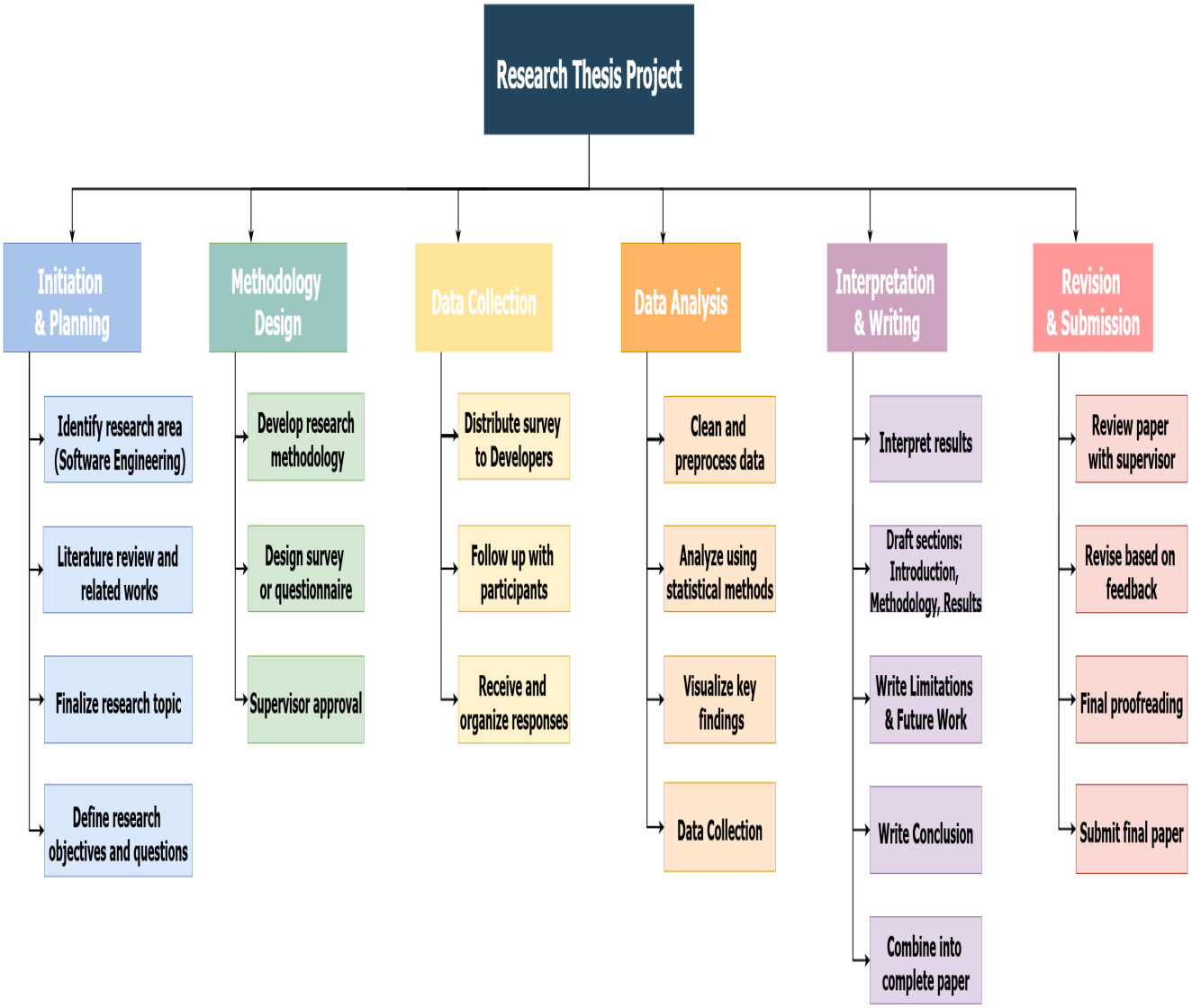

Figure 3.1: Work Breakdown Structure

3.4 Gantt Chart

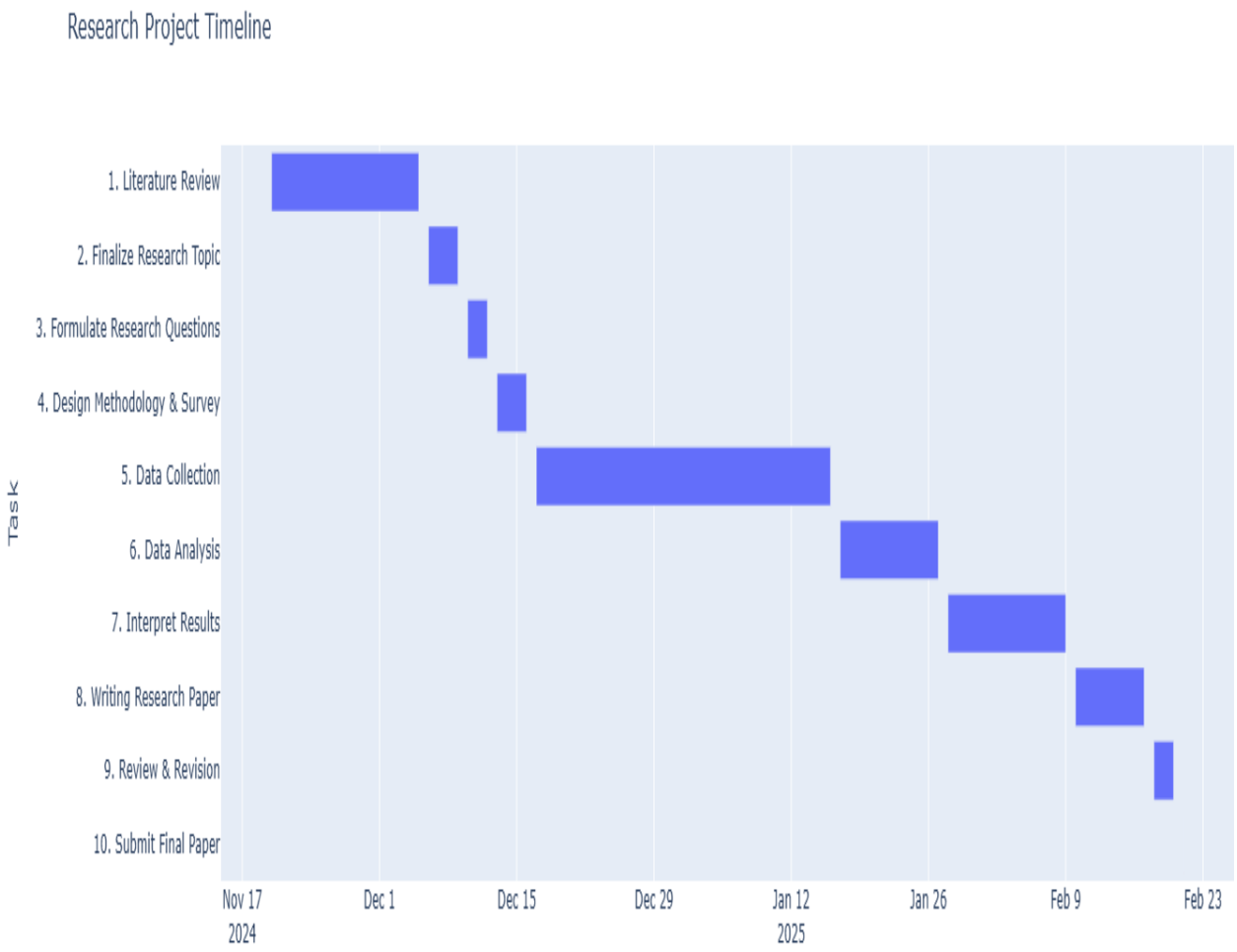

Figure 3.2: Gantt Chart

3.5 Network Diagram

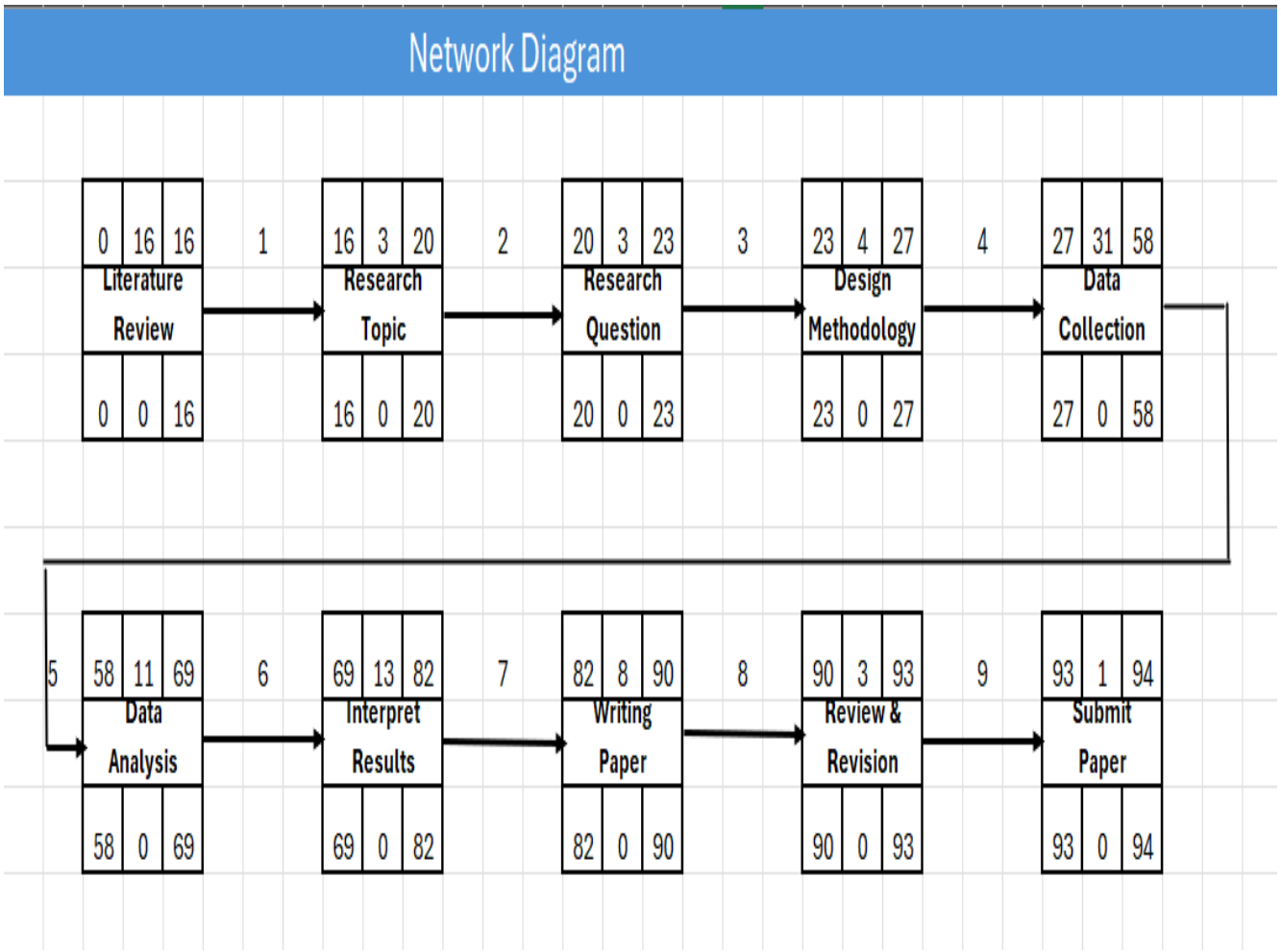

Figure 3.3: Network Diagram

Since every activity has 0 slack, any delay in any task will delay the entire project. Therefore, all 10 tasks are critical. Ensuring that all tasks are completed on time without delay, since the entire project path is critical.

Chapter 4

Investigating the Key Factors Influencing Software Developer Productivity in Developing Regions

4.1 Problem Statement

Despite the increasing importance of software development in emerging economies like Bangladesh, there remains a lack of context-specific understanding of the factors that influence software developer productivity. Existing research tends to focus on global or corporate-scale trends, often overlooking the unique challenges faced by developers in smaller or medium-sized teams operating in developing regions. These challenges include varied work environments, emotional well-being, access to tools, and organizational culture—all of which can significantly affect productivity. Without localized insights, organizations in Bangladesh struggle to implement effective, evidence-based strategies to support and enhance developer performance.

4.2 Research Methodology

The purpose of this paper was to find out the key factors that affect the productivity of software developers. To achieve this, a survey-based approach was adopted, incorporating a range of validated productivity factors. This method was chosen because other quantitative approaches—such as measuring lines of code or the number of bugs fixed—do not reliably capture developer productivity, as developers often spend a significant portion of their time in meetings and requirement gathering sessions [8].

Alternative methods such as literature reviews, diary studies, peer evaluations, interviews, and biometric monitoring also have notable limitations. Interviews and diary studies can be time-consuming and impractical for large-scale participation. Biometric sensors may introduce response bias, as participants are aware of being monitored, and such tools may require individual calibration [7]. Peer reviews may introduce bias due to interpersonal dynamics. In contrast, surveys offer a straightforward and scalable method of data collection, allowing for

efficient analysis across a broad participant base. These characteristics make the survey method particularly suitable and reliable for this study.

The study specifically focuses on identifying productivity factors in the context of Bangladesh's software industry. It addresses the following core research questions: (1) What are the primary factors affecting the productivity of developers in Bangladesh? (2) How do these factors vary across different companies? and (3) What predicts productivity across experience levels? By addressing these questions, the research provides insights into how individual, organizational, and experiential variables influence developer productivity. These findings aim to inform actionable strategies for improving both productivity and job satisfaction in the region.

The methodological foundation of this study is based on the validated framework developed by Murphy-Hill et al. [61], which has been empirically tested in large-scale studies. Their original list of 127 candidate factors was refined to 48 through a rigorous process involving the elimination of duplicates, consolidation of overlapping themes, and prioritization of factors with clear practical relevance. The final selection was reviewed iteratively by industry experts at Google, ensuring alignment with real-world software development environments. These 48 factors were then divided into few categories like - work environment, technical skills, technology and tools.

4.2.1 Purpose of the Study

The purpose of this study is to find the key factors affecting developer productivity in developing regions like Bangladesh. By surveying developers across 38 software firms, this study provides insights and suggestions that organizations may use to improve their developers' productivity and hence produce a better profit and outcome for their organization.

4.2.2 Data Collection

Data was collected using an online survey. The survey was sent to 168 developers across 38 different software company with 2-3 developers from each company. The survey was conducted over a period of approximately one month. Participation was entirely voluntary, and the survey description clearly stated that all responses would remain confidential and used solely for academic research purposes. Informed consent was implied through voluntary participation.

The survey began with general demographic questions such as primary role, mode of employment, and years of professional experience, followed by 48 validated productivity factors rated using a five-point Likert scale (Strongly Disagree to Strongly Agree). These factors were adopted from Murphy-Hill et al. (2019) to ensure consistency and methodological rigor. An open-ended question was also included at the end to allow respondents to identify any additional productivity factors not covered in the closed items.

A total of 44 valid responses were collected, resulting in a response rate of 26%, which aligns with typical response rates in software engineering survey research.

4.2.3 Data Analysis

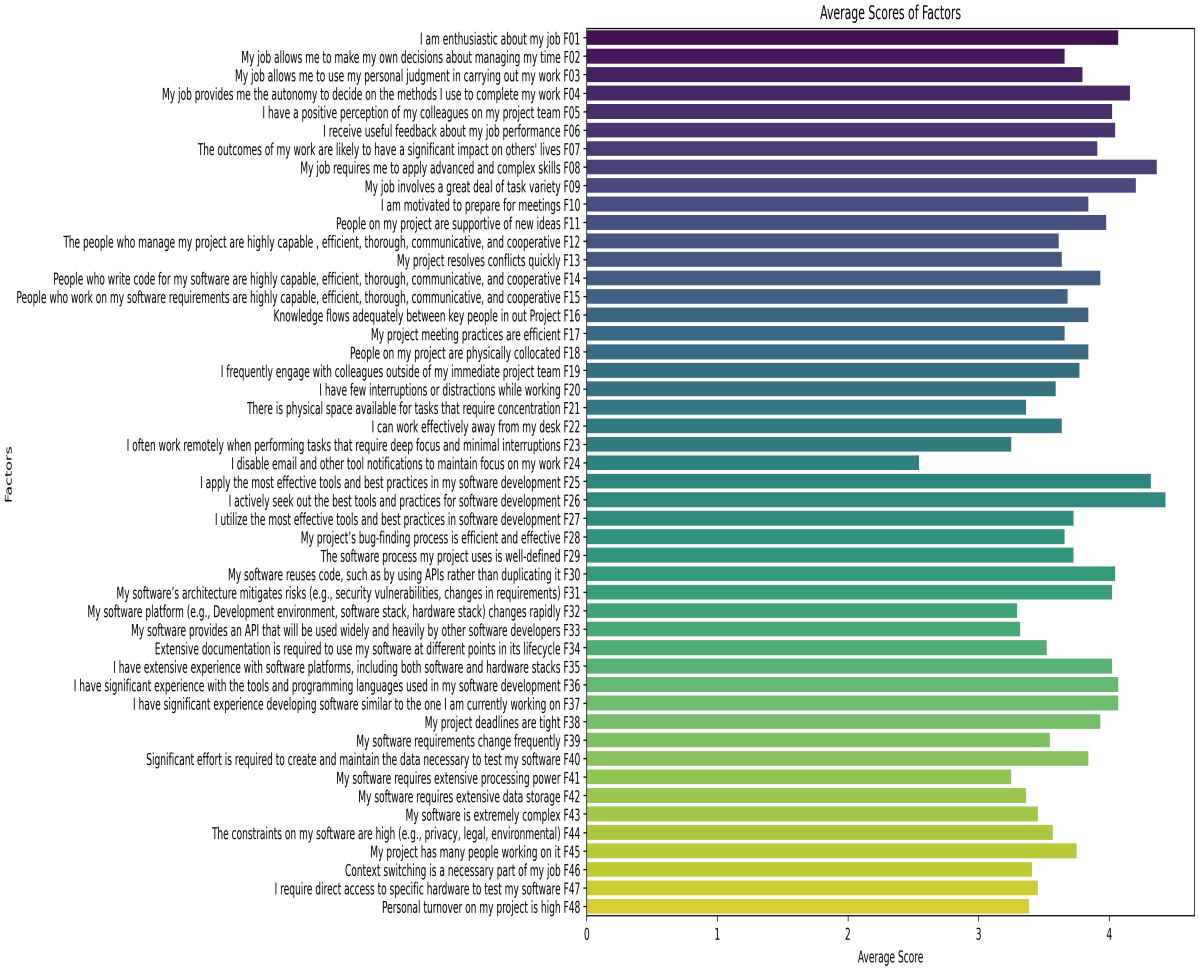

Figure 4.1: Bar Chart representing average scores of factors influencing software developer productivity

Figure 4.1 presents the mean scores of all 48 productivity factors. The left-hand side lists the statements shown to the developers (e.g., F1, F2, F3), while the right-hand side displays their corresponding average values. The mean for each factor was calculated by summing all responses for that item and dividing by the total number of respondents. Factors with the highest average scores indicate the strongest perceived influence on developer productivity, while those with the lowest scores are considered the least influential.

To explore the variability in responses, the standard deviation of each of the 48 factors was also calculated. Figure 4.2 illustrates the dispersion of responses, where a lower standard deviation indicates consensus among participants, and a higher value suggests diverse opinions. This helps identify which productivity factors are more universally agreed upon versus those that are perceived differently across respondents.

To assess how perceptions of productivity factors vary by experience, participants were categorized into three groups: Beginners (0–2 years), Mid-level (3–5 years), and Senior developers

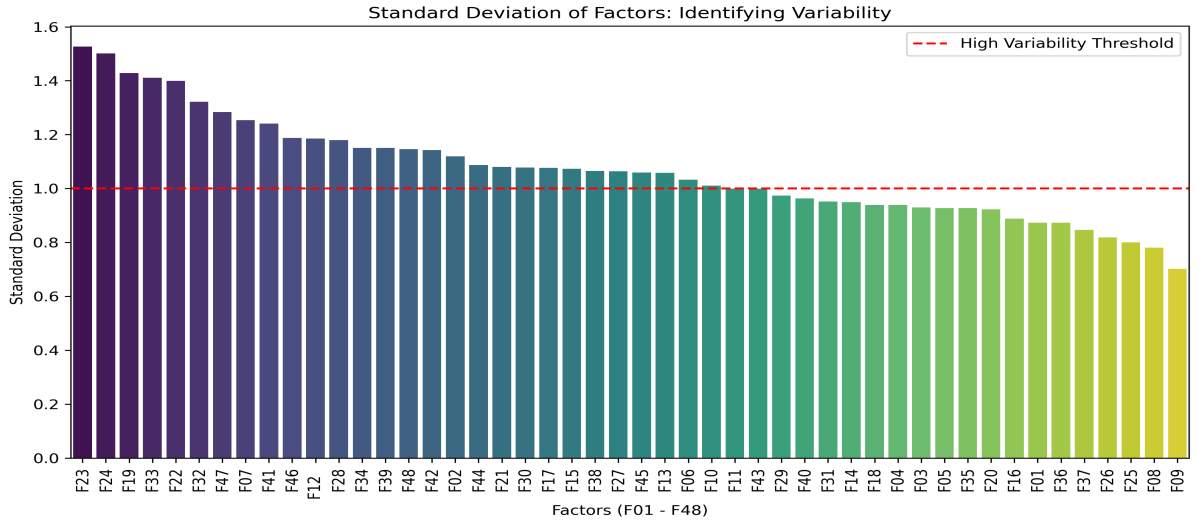

Figure 4.2: Standard Deviation of the factors influencing software developer productivity

Figure 4.3: Heatmap showing the mean scores of productivity-related factors across different experience levels

(6+ years). The mean score of each factor was calculated within these groups and visualized in a heatmap (Figure 4.3). This visualization reveals how experience influences the prioritization of productivity-related factors among developers in Bangladesh.

4.2.4 Research Method

A structured online survey was used to collect primary data. The survey instrument consisted of 48 productivity factors, each measured on a five-point Likert scale ranging from "Strongly Disagree" to "Strongly Agree." These items were drawn from the validated list proposed by [61]. Additionally, an open-ended question was included at the end of the survey, allowing respondents to highlight any factors that may have been overlooked. A total of 168 invitations was sent to software developers across 38 different software firms, with 3-4 developers invited from each organization. The survey remained open for approximately one month. The survey description explicitly mentioned that all the information provided, including any identifiable de tails, will remain completely confidential and will be used for research purposes and that the participation is entirely voluntary. Initially the survey started with Participants name, Primary role, Mode of Employment, Years of professional experience, followed by the 48 factors on a five-point scale, from Strongly disagree to Strongly agree. At the end of the survey, an open-ended question allowed respondents to highlight any additional factors that may be overlooked. The Primary Role question, included predefined options such as Software Engineering, Front-end Developer,Back-end Developer, Full-Stack Developer, DevOps Engineer and an "Other" option for additional roles. The mode of Employment question includes Full-time(Onsite), Full-time(Hybrid), Full-time(Remote), Part-time, Freelance and "Other" option. For profession experience respondents chose from the following ranges: 0-2 years, 3-5 years, 6-10 years, 11-15 years, 16+. A total of 44 responses were collected. The response rate is 26

A structured online survey was used to collect primary data for this study. The survey instrument included 48 productivity factors, each evaluated using a five-point Likert scale, ranging from "Strongly Disagree" to "Strongly Agree". These items were drawn from the validated framework proposed by [61]. Additionally, an open-ended question was included at the end of the survey, allowing respondents to highlight any factors that may have been overlooked. A total of 168 invitations were sent to software developers working in 38 software firms across Bangladesh, with 3–4 developers invited from each organization. The survey remained open for approximately one month. The survey description explicitly mentioned that all the information provided, including any identifiable details, will remain completely confidential and will be used for research purposes and that the participation is entirely voluntary. The survey began with demographic questions, including: Participant's Name (optional), Primary Role, Mode of Employment, and Years of Professional Experience. This was followed by the 48 productivity factor items. For Primary Role, participants selected from predefined options such as Software Engineer, Front-End Developer, Back-End Developer, Full-Stack Developer, DevOps Engineer, or chose "Other". The Mode of Employment included: Full-time (Onsite), Full-time (Hybrid), Full-time (Remote), Part-time, Freelance, or "Other".

Participants also categorized their professional experience into one of five groups: 0–2 years,

3–5 years, 6–10 years, 11–15 years, and 16+ years. A total of 44 valid responses were collected, resulting in a response rate of 26%, which aligns with typical return rates reported in similar software engineering productivity studies [1].

4.3 Proposed Solution

4.3.1 Your Proposed Solutions

The results of this study suggest several practical steps that organizations in Bangladesh can take to improve the productivity of their software developers. These solutions are intended to support both developer performance and overall organizational outcomes.

- **Promote Task Variety:** Giving developers a mix of tasks can keep them motivated and mentally engaged. Rotating assignments and avoiding repetitive work can help reduce monotony and improve focus.
- **Support Skill Development and Problem Solving:** Developers should be encouraged to learn continuously and work on tasks that require problem-solving. Providing access to training materials, mentorship, and challenging projects can strengthen their technical abilities.
- **Encourage Autonomy:** Allowing developers some freedom in choosing tools and methods can increase their sense of responsibility. Greater independence often leads to higher efficiency and satisfaction with their work.
- **Adopt Modern Tools and Practices:** Using updated technologies, frameworks, and industry best practices makes development smoother and less time-consuming. Regular training sessions can help teams stay familiar with new tools.
- **Match Strategies to Experience Level:** Since developers' needs change with experience, organizations should adapt their strategies. For instance, junior developers may need structured guidance, while senior developers may be more effective in leadership or mentoring roles.
- **Adapt Workplace Policies:** A supportive work environment is important, whether teams are working remotely, in hybrid settings, or onsite. Reducing unnecessary meetings, ensuring fair workloads, and building a culture of cooperation can improve productivity.

These recommendations highlight practical ways to strengthen developer productivity. By implementing them, organizations can create a more supportive and effective work environment for their software teams.

4.4 Result Analysis

This section, the survey data is analyzed to answer the three research questions. The Cronbach's Alpha value for the survey came out to be 0.931, suggesting excellent reliability and internal consistency.

4.4.1 RQ1: Primary Factors Affecting Developer Productivity

The primary factors affecting developer productivity are:

- My job provides flexibility in choosing the method I use to complete my task (F04)
- Complex and high level skill required to complete task (F08)
- My job involves a deal of task variety (F09)
- I actively seek out the best tools and practices to develop my software (F26)
- I utilize the most effective tools and best practices in software development (F27)

These factors suggest that both task-related and technical aspects significantly enhance productivity. Developers value autonomy and intellectual challenge in their tasks, while also relying on modern tools to optimize performance. This aligns with existing literature but also highlights regional nuances, such as a lower impact of remote work compared to studies from Western contexts.

4.4.2 RQ2: Variation Across Companies

The variability of productivity factors across different companies was examined through standard deviation analysis. The most stable factors across organizations were:

- Use of best tools and practices (F25)
- Requirement of complex skills (F08)
- Task variety (F09)
- In contrast, the most variable factors were:
- Remote work during focused tasks (F23)
- Managing notifications to maintain focus (F24)
- Cross-team collaboration (F19)

These findings indicate that while technical efficiency is consistently emphasized across companies, workplace policies regarding flexibility and communication vary significantly.

4.4.3 RQ3: Productivity and Experience Level

Participants were grouped by experience: beginners (0–2 years), mid-level (3–5 years), and seniors (6+ years). The analysis showed:

- Beginners prioritize tool usage and learning complex skills.
- Mid-level developers are most affected by tight deadlines and challenging tasks.
- Senior developers emphasize leadership, decision-making autonomy, technical expertise, and collaboration.

This progression reflects a shift in priorities from individual skill development to strategic project and team management as experience increases.

4.4.4 Additional Insights from Open-Ended Responses

Respondents also highlighted additional productivity factors not covered in the closed-ended survey, such as:

- Legacy systems and outdated tools
- Flexible working hours
- Communication gaps with clients
- Work-life balance
- Toxic work environments
- Multitasking across projects
- Overtime and burnout

These qualitative insights reinforce the importance of organizational and psychological factors in shaping developer productivity and suggest areas for managerial improvement.

4.4.5 Demographic Results

This section presents the demographic distribution of the respondents who participated in the survey. A total of 44 valid responses were collected from software developers working in 38 different software firms across Bangladesh. Participants were asked to provide basic background information such as their primary job role, mode of employment, and years of professional experience.

- **Experience Level:**
  - 0–2 years: 45.5%

  - 3–5 years: 29.5%
  - 6–10 years: 13.6%
  - 11–15 years: 9.1%
  - 16+ years: 2.3%

- **Primary Roles:** Respondents held various positions such as:
  - Software Engineer
  - Front-end Developer
  - Back-end Developer
  - Full-stack Developer
  - DevOps Engineer
  - Other (as specified by the participant)

- **Mode of Employment:**
  - Full-time (Onsite)
  - Full-time (Hybrid)
  - Full-time (Remote)
  - Part-time
  - Freelance
  - Other

All respondents were informed about the confidentiality and voluntary nature of the survey. This demographic breakdown helps contextualize the results and ensures the inclusion of a diverse group of developers, reflecting a broad view of the software development workforce in Bangladesh.

Figures 4.4 and 4.5 visually represent the distribution of employment types and job roles, respectively.

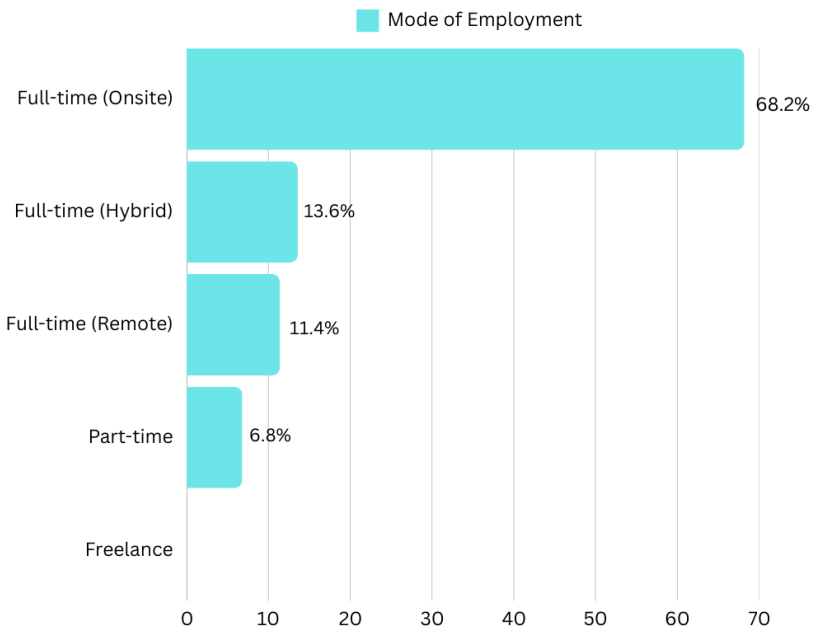

Figure 4.4: Distribution of Respondents by Employment Mode

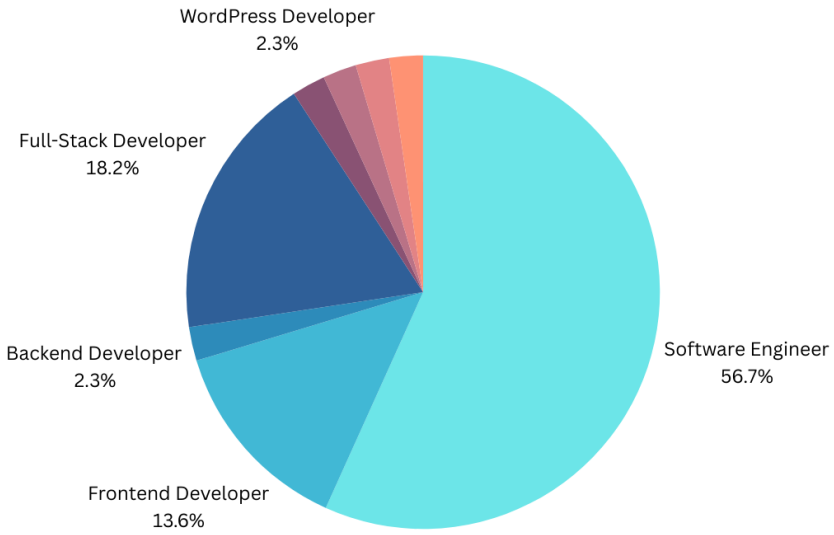

Figure 4.5: Distribution of Respondents by Job Role

4.4.6 Exploratory Analysis

This part of the study focused on reviewing the survey data to look for patterns and trends among the 48 factors that may influence software developer productivity. The purpose was to gain an early understanding of the data before moving to detailed result analysis.

First, the mean scores of all 48 factors were calculated. A bar chart was prepared to show these averages, which made it easier to see which factors were rated the highest overall. Task

variety, the use of modern tools and practices, problem-solving opportunities, and freedom in choosing work methods received the top ratings, suggesting that these elements strongly affect productivity.

Next, the standard deviation of each factor was examined to see how much agreement there was among participants. A low standard deviation showed that responses were fairly consistent, while a high value indicated mixed opinions. This step helped in identifying where developers agreed most and where their views differed, possibly due to differences in workplace context.

Finally, the data was grouped based on developer experience. Three categories were used: beginner (0–2 years), mid-level (3–5 years), and senior (6+ years). A heatmap was created to compare how these groups rated the factors. The results showed that beginner developers focused more on learning and adapting to tools, mid-level developers were more affected by deadlines and task complexity, while senior developers placed greater importance on leadership, teamwork, and autonomy. These differences highlight how priorities change as developers gain experience.

Overall, this exploratory stage provided useful insights and set the foundation for the more detailed analysis that follows in the results section.

Chapter 5

Sustainability

5.1 Sustainability Overview

Sustainability for software engineering takes into account the economic, social and environmental sustainability. The findings of this research contribute to building more resilient, inclusive, and efficient software development practices. By identifying key productivity factors and recommending actionable strategies, the study promotes organizational well-being and individual job satisfaction, ensuring that improvements are not just short-lived but embedded into the structural functioning of software firms. This chapter explores the economic, social, environmental, and ethical implications of the research.

5.2 Economical Sustainability

The research contributes to economic sustainability by offering insights that can help optimize human capital within software organizations. In a growing digital economy like Bangladesh, improved developer productivity can directly enhance software delivery timelines, reduce project costs, and increase return on investment. By addressing inefficiencies and promoting factors such as autonomy, task variety, and tool usage, the proposed solutions can help companies maintain competitiveness without excessive resource expenditure. Moreover, by aligning work conditions with developer expectations, firms can reduce turnover rates and the financial burden associated with recruitment and on-boarding.

Cost	Year 1	Year 2	Year 3	Year 4	Year 5
Software Licenses Tk	306735	322071.75	338175.34	355084.10	372838.31
AI Tool Tk	544500	571725	600311.25	630326.81	661843.15
Training Tk	182000	191100	200655	210687.75	221222.14
Total Cost Tk	1033235	1084896.75	1139141.59	1196098.67	1255903.60
Cumulative Cost Tk	1033235	2118131.75	3257273.34	4453372.00	5709275.61
Benefits					
Developer Salary Tk	5400000	5940000	6534000	7187400	7906140
Total Benefit Tk	972000	1069200	1176120	1293732	1423105.20
Cumulative Benefit Tk	972000	2041200	3217320	4511052	5934157.20
NPV	-61235	-76931.75	-39953.34	57680	224881.60
ROI (Total Project)	4.11%				

Table 5.1: Cost-Benefit Analysis over 5 Years

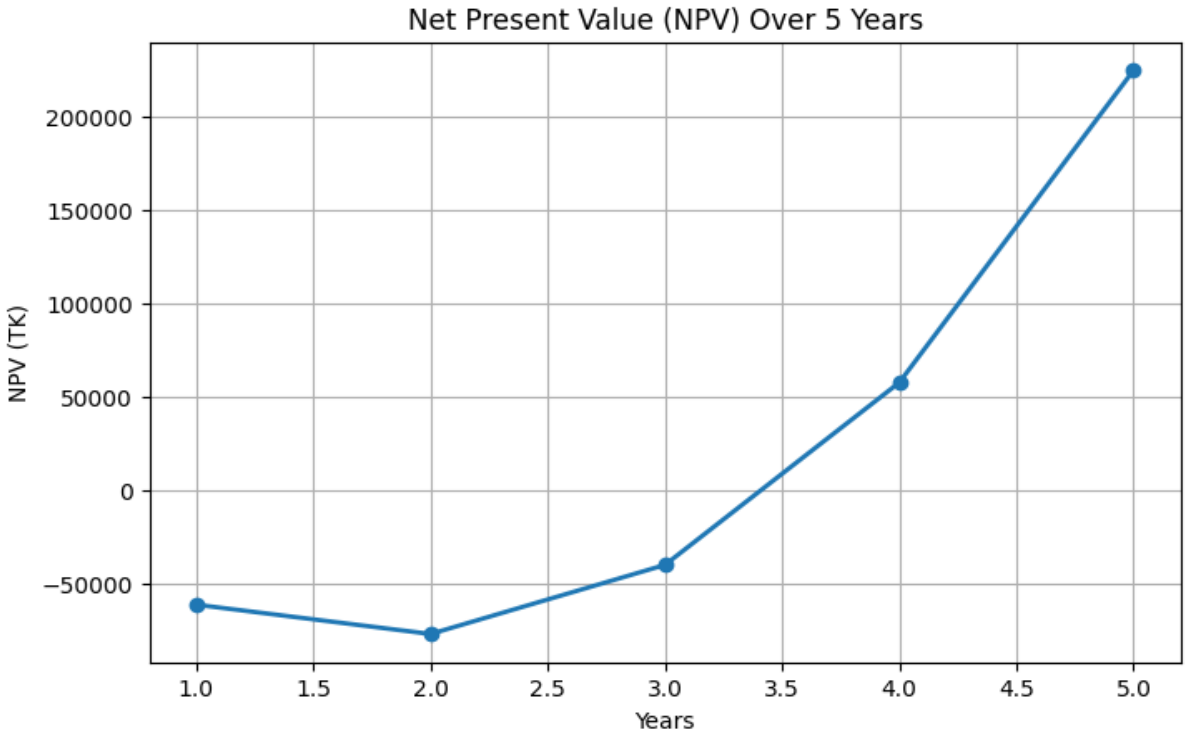

Figure 5.1: NPV Analysis of the Project

Since this thesis focuses on software productivity, the proposed solution suggests incorporating paid IDEs, AI-based tools, and training developers in new and emerging programming languages and frameworks. By adopting these tools and practices, organizations can achieve faster development cycles, reduced debugging time, higher output per developer, and lower overtime costs.

Considering that most Bangladeshi local firms typically operate with small teams of 10–15 developers, we assumed a team size of 15 developers for our cost-benefit estimation.

- **Software Licenses:** We considered IntelliJ IDEA with an annual cost of $169 per user. This was multiplied by 15 developers and then converted to Bangladeshi Taka (BDT). Additionally, we assumed a 5% annual increase in subscription costs.
- **AI Tools:** For AI support, we used ChatGPT Plus with an annual subscription fee per user. The cost was multiplied by 15 developers, converted to BDT, and assumed to increase by 5% annually.
- **Training Costs:** For developer upskilling, we assumed an average course cost of $100 per developer (e.g., Udemy courses). This was multiplied by 15 developers and converted to BDT.
- **Total Cost:** The total cost was then calculated as the sum of software licenses, AI tools, and training costs:

$$\text{Total Cost} = \text{Software Licenses} + \text{AI Tools} + \text{Training Costs}$$

- **Developer Salary and Benefits:** Each developer's monthly salary was assumed to be 30,000 BDT. To calculate annual salary expenses, we multiplied by 12 (months) and then by 15 developers:

$$\text{Annual Salary} = 30{,}000 \times 12 \times 15$$

  For benefits, we assumed an additional 20% of the total developer salary:

$$\text{Benefits} = 0.18 \times \text{Annual Salary}$$

- **NPV, and ROI:** Finally, the NPV was computed as:

$$\text{NPV} = \text{Total Benefits} - \text{Total Cost}$$

  Using this, we further calculated the Net Present Value (NPV) and Return on Investment (ROI) following standard financial formulas:

$$ROI = \frac{\text{Total Benefits-Total Cost}}{\text{Total Cost}} \times 100\%$$

The ROI for the entire project over a 5-year period was calculated to be 4.11%.

5.3 Social and Environmental Sustainability

From a social perspective, this research advocates for a work environment that supports employee well-being, job satisfaction, and career development. Recognizing the role of emotional health, learning opportunities, and work-life balance, the study supports practices that foster a positive organizational culture. These social improvements can lead to higher retention, better collaboration, and more inclusive workplaces.

While the direct environmental impact of the research is minimal, the promotion of remote and hybrid work models (as analyzed in the survey) indirectly supports environmental sustainability. By reducing the need for daily commuting and large-scale office infrastructure, software firms can lower their carbon footprint. Encouraging digital collaboration and cloud-based workflows aligns with broader goals of sustainable digital transformation.

5.4 Ethics

Ethical considerations were integral to every phase of this research. Participants were informed about the confidentiality of their responses and the voluntary nature of the survey. No personal identifiers were used in the analysis, and data was handled with full respect for privacy and academic integrity. Furthermore, the study avoids any form of bias or discrimination by ensuring diverse participation across different firms, roles, and experience levels. The ethical focus of this paper is that it promotes autonomy, respects developer perspective, and upholds equality in workplace.

Chapter 6

Conclusion

6.0.1 Project Summary

This study examined the key factors influencing software developer productivity in Bangladesh, focusing on three main aspects: the strongest productivity factor (RQ1), how do these factors vary between different companies (RQ2) and how productivity factors differ across experience levels (RQ3). This study found that task variety (F09), seeking out and using the best tools and practices in development (F26 and F27), allowing developers to chose their method to complete work (F04) and high-level skill and complex problem solving (F08) as the primary factors affecting developer in Bangladesh. We analyzed how productivity factors vary across different companies and this study found task variety (F09) as the most stable factor. Finally we investigate how productivity factors evolve with experience. Beginner developers prioritize learning and skill development, actively seeking out best practices and modern tools. Mid-level developers, on the other hand, face increasing pressure from tight deadlines and the complexity of their tasks. Senior developers emphasize leadership, teamwork, and decision-making autonomy, suggesting that experience shifts priorities from technical execution to strategic contribution and collaboration. Understanding what makes software developer productive is crucial for both individual career growth and organizational success. By identifying key productivity factors and analyzing how they differ across companies and experience levels, this study provides actionable insights that can help improve developer efficiency, job satisfaction, and workplace policies in Bangladesh's growing software industry.

6.0.2 Future Work

This study offers a valuable snapshot of how job satisfaction and productivity factors vary across different experience levels of software developers. However, several areas remain open for future research.

One potential direction is to conduct longitudinal studies that track developers over time, examining how their priorities, challenges, and productivity-related factors evolve as they gain experience. Such an approach would provide deeper insight into developer growth trajectories beyond cross-sectional data.

As this research relies on self-reported survey responses, it may be subject to subjective bias. Future studies could address this limitation by integrating objective productivity metrics—such as GitHub commit frequency, code quality assessments, or bug resolution rates—to complement self-perception data.

Furthermore, while this study considered experience levels, it did not incorporate job roles in the analysis. Future research could explore the relationship between job role variation (e.g., front-end, back-end, DevOps) and perceived productivity.

The influence of remote and hybrid work arrangements also warrants deeper investigation. As flexible work models become increasingly common, understanding their specific impacts on productivity in the software industry—especially in regional contexts like Bangladesh—would be highly beneficial.

Lastly, the impact of artificial intelligence (AI) tools on developer productivity is an emerging area of critical importance. With the rise of AI-powered tools such as GitHub Copilot and ChatGPT, future studies should explore how these technologies reshape workflows, productivity, and even developer roles.

In summary, addressing these areas in future research could enhance and build upon the findings of this study, contributing to a more comprehensive understanding of the evolving nature of software development productiv

Bibliography

[1] B. Johnson, T. Zimmermann, and C. Bird, "The effect of work environments on productivity and satisfaction of software engineers," *IEEE Transactions on Software Engineering*, vol. 47, no. 4, pp. 736–757, 2019.

[2] L. Bao, T. Li, X. Xia, K. Zhu, H. Li, and X. Yang, "How does working from home affect developer productivity?—a case study of baidu during the covid-19 pandemic," *Science China Information Sciences*, vol. 65, no. 4, p. 142102, 2022.

[3] A. Ruvimova, A. Lill, J. Gugler, L. Howe, E. Huang, G. Murphy, and T. Fritz, "An exploratory study of productivity perceptions in software teams," in *Proceedings of the 44th International Conference on Software Engineering*, pp. 99–111, 2022.

[4] A. N. Meyer, "Fostering software developers' productivity at work through self-monitoring and goal-setting," in *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*, pp. 480–483, 2018.

[5] I. Obi, J. Butler, S. Haniyur, B. Hassan, M.-A. Storey, and B. Murphy, "Identifying factors contributing to bad days for software developers: A mixed methods study," *arXiv preprint arXiv:2410.18379*, 2024.

[6] L. Cheng, E. Murphy-Hill, M. Canning, C. Jaspan, C. Green, A. Knight, N. Zhang, and E. Kammer, "What improves developer productivity at google? code quality," in *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 1302–1313, 2022.

[7] D. Girardi, F. Lanubile, N. Novielli, and A. Serebrenik, "Emotions and perceived productivity of software developers at the workplace," *IEEE Transactions on Software Engineering*, vol. 48, no. 9, pp. 3326–3341, 2021.

[8] M.-A. Storey, T. Zimmermann, C. Bird, J. Czerwonka, B. Murphy, and E. Kalliamvakou, "Towards a theory of software developer job satisfaction and perceived productivity," *IEEE Transactions on Software Engineering*, vol. 47, no. 10, pp. 2125–2142, 2019.

[9] D. Smit, H. Smuts, P. Louw, J. Pielmeier, and C. Eidelloth, "The impact of github copilot on developer productivity from a software engineering body of knowledge perspective," 2024.

[10] M. Coutinho, L. Marques, A. Santos, M. Dahia, C. França, and R. de Souza Santos, "The role of generative ai in software development productivity: A pilot case study," in *Proceedings of the 1st ACM International Conference on AI-Powered Software*, pp. 131–138, 2024.

[11] F. Ibrahim, Y. E. Ashfaque, M. Roknuzzaman, M. Islam, S. J. Sumi, N. Nahar, and M. Hasan, "Bridging the security gap in sdlc: Challenges and solutions for local it firms in bangladesh," in *EPJ Web of Conferences*, vol. 328, p. 01046, EDP Sciences, 2025.

[12] M. M. Hasan, M. A. Rahman, M. S. Chowdhury, M. H. Rahman, K. H. Abdulle, F. Sadia, and M. Hasan, "Testing react single page web application using automated testing tools.," in *ENASE*, pp. 469–476, 2022.

[13] M. M. Morshed, M. Hasan, and M. Rokonuzzaman, "Software architecture decision-making practices and recommendations," in *Advances in Computer Communication and Computational Sciences: Proceedings of IC4S 2018*, pp. 3–9, Springer, 2019.

[14] M. Islam, F. Khan, S. Alam, and M. Hasan, "Artificial intelligence in software testing: A systematic review," in *TENCON 2023-2023 IEEE Region 10 Conference (TENCON)*, pp. 524–529, IEEE, 2023.

[15] M. Islam, F. Khan, M. Hasan, F. Sadia, and M. Hasan, "Impact of covid-19 on the factors influencing on-time software project delivery: An empirical study," 18th International Conference on Evaluation of Novel Approaches to Software ..., 2023.

[16] L. N. Raha, A. W. Hossain, T. Faiyaz, M. Hasan, N. Nahar, and M. Rokonuzzaman, "A guide for building the knowledgebase for software entrepreneurs, firms, and professional students," in *2018 IEEE 16th International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 165–171, IEEE, 2018.

[17] A. Sultana, M. Islam, M. Hasan, and F. Ahmed, "Fake news detection using machine learning techniques," in *2023 IEEE/ACIS 21st International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 98–103, IEEE, 2023.

[18] A. Ahamed, N. Sadman, T. A. Khan, M. I. Hannan, F. Sadia, and M. Hasan, "Automated testing: Testing top 10 owasp vulnerabilities of government web applications in bangladesh," *ICSEA 2022*, vol. 56, 2022.

[19] S. Saha, M. A. B. Khaled, M. S. Islam, N. S. Puja, and M. Hasan, "Detecting sex from handwritten examples," in *2018 ieee international conference on system, computation, automation and networking (icscan)*, pp. 1–7, IEEE, 2018.

[20] M. N. Mim, M. Firoz, M. M. Islam, M. Hasan, and M. T. Habib, "A study on social media addiction analysis on the people of bangladesh using machine learning algorithms," *Bulletin of Electrical Engineering and Informatics*, vol. 13, no. 5, pp. 3493–3502, 2024.

[21] F. Oyshi, A. Bonik, M. O. H. Shin, J. Rashid, M. Akter, M. Hasan, and F. Sadia, "A novel model to adapt cmmi level 2 by assessing the local smes of bangladesh," *Procedia Computer Science*, vol. 219, pp. 2043–2050, 2023.

[22] S. Irtisamul, M. B. Hassan, F. Sadia, M. Hasan, and M. Rokonuzzaman, "Understanding consumer perceptions of green software: A study in the online social media marketing context," *Procedia Computer Science*, vol. 237, pp. 775–782, 2024.

[23] A. Sultana, M. Hasan, M. Shidujaman, and C. Premachandra, "Sentiment analysis with deep learning methods for performance assessment and comparison," in *2024 International Conference on Image Processing and Robotics (ICIPRoB)*, pp. 1–6, IEEE, 2024.

[24] F. Abdullah, M. A. Rahman, M. Shidujaman, M. Hasan, and M. T. Habib, "Machine learning modeling for reconditioned car selling price prediction," in *Seventh International Conference on Mechatronics and Intelligent Robotics (ICMIR 2023)*, vol. 12779, pp. 605–612, SPIE, 2023.

[25] M. Islam and M. Hasan, "Big data analysis driven decision making system ensuring energy security of a country," in *Proceedings of the 2021 7th International Conference on Computer Technology Applications*, pp. 60–65, 2021.

[26] M. Shafir, P. P. Saha, A. T. Araf, J. F. Nishi, M. Hasan, and F. Sadia, "The success factors of agile methodologies in software development based on developing countries' software firms," *Procedia Computer Science*, vol. 256, pp. 1954–1961, 2025.

[27] M. T. Hasan, F. Sadia, M. Hasan, and M. Rokonuzzaman, "Develop a system to analyze logs of a given system using machine learning," in *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing: Volume 17*, pp. 1–13, Springer, 2024.

[28] M. T. Hasan, M. Akter, S. I. Shuhrid, T. A. Khan, F. Sadia, and M. Hasan, "Investigation the influence of marketing-mix efforts on brand equity in the bangladesh software industry.," in *ICSBT*, pp. 152–159, 2022.

[29] M. M. Alam, S. I. Priti, K. Fatema, M. Hasan, and S. Alam, "Ensuring excellence: A review of software quality assurance and continuous improvement in software product development," *Achieving sustainable business through AI, technology education and computer science*, pp. 331–346, 2024.

[30] S. R. Akthar, F. Sadia, and M. Hasan, "Strategies to overcome slow growth in software firms: Project diversity perspective," tech. rep., tech. rep., EasyChair, 2023.

[31] F. Sadia, M. Hasan, N. Nahar, and M. Rokonuzzaman, "A new process model of incremental asset building for software project management," in *2021 IEEE/ACIS 19th International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 86–90, IEEE, 2021.

[32] M. T. Hasan, N. M. A. Bakar, N. Nahar, M. Hasan, and M. Rokonuzzaman, "A hybrid approach to overcome requirements challenges in the software industry," 2023.

[33] S. A. Lesum, S. R. Akthar, M. R. Islam, F. Sadia, and M. Hasan, "Project governance to improve the performance of software projects by mitigating the software risk factors: The moderating role of project leadership," *Procedia Computer Science*, vol. 239, pp. 1863–1870, 2024.

[34] M. Hasan and M. A. Cheema, "Efficient algorithms for furthest k neighbors queries in spatial databases," in *13th International Conference on Computer and Information Technology (ICCIT)*, 2010.

[35] M. M. Alam, S. I. Priti, K. Fatema, M. Hasan, and S. Alam, "Cmmi in practice: Enhancing organizational performance through holistic approaches," *Achieving Sustainable Business Through AI, Technology Education and Computer Science: Volume 3: Business Sustainability and Artificial Intelligence Applications*, pp. 15–28, 2024.

[36] M. T. Hasana, F. Akterb, S. Alamc, and M. Hasand, "Quality assurance model for lms to facilitate education and equality,"

[37] S. Shikta, S. K. Das, S. N. Mahal, H. M. Shahriyar, K. B. Al Jannat, and M. Hasan, "Requirement engineering in startups.," in *ICSOFT*, pp. 207–214, 2021.

[38] T. T. Mayabee, S. Khan, A. Alam, S. Amin, J. K. Chowdhury, M. T. Hassan, S. Alam, and M. Hasan, "Student performance monitor: A big data analytical application," in *Proceedings of International Conference on Data Science and Applications: ICDSA 2021, Volume 2*, pp. 759–771, Springer, 2021.

[39] A. Sarah, M. I. H. Rabbi, M. S. Siddiqua, S. Banik, and M. Hasan, "Prediction of cumulative grade point average: A case study," in *Science and Information Conference*, pp. 33–42, Springer, 2020.

[40] T. Z. Khan, S. H. Tusher, M. Hasan, and M. Rokonuzzaman, "Tailoring scrum methodology for game development," in *Advances in Computer, Communication and Computational Sciences: Proceedings of IC4S 2019*, pp. 233–243, Springer, 2020.

[41] J. Ferdouse, M. A. A. Siddique, M. S. Haque, M. Hasan, and M. Tarek, "Intelligent systems and applications in engineering,"

[42] F. F. Bidushi, Z. B. Jashim, M. K. Anik, M. Islam, and M. Hasan, "Banking software services: Current status, challenges, impact and prospects," in *Information and Communication Technology for Competitive Strategies (ICTCS 2022) Intelligent Strategies for ICT*, pp. 123–133, Springer, 2023.

[43] M. S. Islam, T. Islam, and M. Hasan, "Approaching deep convolutional neural network for biometric recognition based on fingerprint database," in *Intelligent Computing: Proceedings of the 2021 Computing Conference, Volume 2*, pp. 581–590, Springer, 2021.

[44] F. A. Anni, M. R. Islam, F. Sadia, M. Hasan, and M. Rokonuzzaman, "Green computing adoption: Understanding the role of individual, social, and organizational factors.," in *ENASE*, pp. 529–536, 2024.

[45] A. W. Hossain, L. N. Raha, T. Faiyaz, M. Hasan, N. Nahar, and M. Rokonuzzaman, "Sustainable management strategy for software firms to reduce employee turnover due to freelancing," in *First International Conference on Sustainable Technologies for Computational Intelligence: Proceedings of ICTSCI 2019*, pp. 197–205, Springer, 2019.

[46] S. R. Akthar, M. R. Islam, M. B. Hasan, M. S. Siddiqua, S. I. Haque, J. A. Saad, F. Sadia, and M. Hasan, "Overcoming obstacles in model-driven engineering: Lessons from the software industry,"

[47] S. Wagner and M. Ruhe, "A systematic review of productivity factors in software development," *arXiv preprint arXiv:1801.06475*, 2018.

[48] A. Arpteg, B. Brinne, L. Crnkovic-Friis, and J. Bosch, "Software engineering challenges of deep learning," in *2018 44th euromicro conference on software engineering and advanced applications (SEAA)*, pp. 50–59, IEEE, 2018.

[49] G. Casale, C. Chesta, P. Deussen, E. Di Nitto, P. Gouvas, S. Koussouris, V. Stankovski, A. Symeonidis, V. Vlassiou, A. Zafeiropoulos, *et al.*, "Current and future challenges of software engineering for services and applications," *Procedia computer science*, vol. 97, pp. 34–42, 2016.

[50] S. Porru, A. Pinna, M. Marchesi, and R. Tonelli, "Blockchain-oriented software engineering: challenges and new directions," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pp. 169–171, IEEE, 2017.

[51] A. Fan, B. Gokkaya, M. Harman, M. Lyubarskiy, S. Sengupta, S. Yoo, and J. M. Zhang, "Large language models for software engineering: Survey and open problems," in *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*, pp. 31–53, IEEE, 2023.

[52] A. Rasheed, B. Zafar, T. Shehryar, N. A. Aslam, M. Sajid, N. Ali, S. H. Dar, and S. Khalid, "Requirement engineering challenges in agile software development," *Mathematical Problems in Engineering*, vol. 2021, no. 1, p. 6696695, 2021.

[53] A. Nguyen-Duc, B. Cabrero-Daniel, A. Przybylek, C. Arora, D. Khanna, T. Herda, U. Rafiq, J. Melegati, E. Guerra, K.-K. Kemell, *et al.*, "Generative artificial intelligence for software engineering—a research agenda," *Software: Practice and Experience*, 2025.

[54] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in human behavior*, vol. 51, pp. 915–929, 2015.

[55] E. de Souza Nascimento, I. Ahmed, E. Oliveira, M. P. Palheta, I. Steinmacher, and T. Conte, "Understanding development process of machine learning systems: Challenges and solutions," in *2019 acm/IEEE international symposium on empirical software engineering and measurement (esem)*, pp. 1–6, IEEE, 2019.

[56] K. Elghariani and N. Kama, "Review on agile requirements engineering challenges," in *2016 3rd International conference on computer and information sciences (ICCOINS)*, pp. 507–512, IEEE, 2016.

[57] S. Baltes and P. Ralph, "Sampling in software engineering research: A critical review and guidelines," *Empirical Software Engineering*, vol. 27, no. 4, p. 94, 2022.

[58] G. Gousios, A. Zaidman, M.-A. Storey, and A. Van Deursen, "Work practices and challenges in pull-based development: The integrator's perspective," in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 1, pp. 358–368, IEEE, 2015.

[59] B. Fitzgerald and K.-J. Stol, "Continuous software engineering: A roadmap and agenda," *Journal of Systems and Software*, vol. 123, pp. 176–189, 2017.

[60] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi, and T. Zimmermann, "Software engineering for machine learning: A case study," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pp. 291–300, IEEE, 2019.

[61] E. Murphy-Hill, C. Jaspan, C. Sadowski, D. Shepherd, M. Phillips, C. Winter, A. Knight, E. Smith, and M. Jorde, "What predicts software developers' productivity?," *IEEE Transactions on Software Engineering*, vol. 47, no. 3, pp. 582–594, 2019.