SOFTWARE ENGINEERING FUNDAMENTALS IN INDUSTRIAL AND SYSTEMS ENGINEERING: SURVEY-BASED EVIDENCE OF UTILITY

BUKET İPEK AKBAL

ÖZYEĞİN UNIVERSITY

SEPTEMBER, 2025

SOFTWARE ENGINEERING FUNDAMENTALS IN INDUSTRIAL AND SYSTEMS ENGINEERING: SURVEY-BASED EVIDENCE OF UTILITY

by

Buket İpek Akbal

Thesis
Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

in
Industrial Engineering

Advisor: Prof. Ömer Erhun Kundakcıoğlu

Graduate School of Science and Engineering
Özyeğin University
İstanbul

September, 2025

SOFTWARE ENGINEERING FUNDAMENTALS IN INDUSTRIAL AND SYSTEMS ENGINEERING: SURVEY-BASED EVIDENCE OF UTILITY

Approved by:

Prof. Ö. Erhun Kundakcıoğlu, Advisor
Department of Industrial Engineering
Özyeğin University

Assoc. Prof. Elvin Çoban
Department of Industrial Engineering
Özyeğin University

Assoc. Prof. Mustafa Gökçe Baydoğan
Department of Industrial Engineering
Boğaziçi University

Approval Date: August 4, 2025

DECLARATION OF ORIGINALITY

I hereby declare that I am the sole author of this thesis and that this is the true copy of my thesis, including the final revisions, approved by my thesis committee. All data and information have been obtained, produced, and presented in accordance with the rules of research ethics and principles of academic honesty. As required by these rules, to the best of my knowledge I have acknowledged ideas, thoughts, and any copyrighted material in accordance with the standard referencing rules. I certify that any part of this thesis has not been submitted for a degree or diploma in another educational institution.

Buket İpek Akbal

ABSTRACT

Industrial and Systems Engineering (ISE) traditionally focuses on improving processes and resource efficiency. However, with rapid technological developments, the field now increasingly requires skills in programming, data analysis, and algorithmic thinking. The importance of programming skills, including Object-Oriented Programming (OOP) and database knowledge, has grown as modern ISE increasingly relies on data-driven decision-making and advanced computational tools.

This study examines the role of programming and related technical skills in the education and professional activities of ISE practitioners. Based on survey responses, it sheds light on how well current ISE curricula meet industry needs and whether a stronger focus on technological competencies is required alongside traditional ISE knowledge. These findings are further supported by firsthand instructional insights, highlighting that while the use of more accessible prototyping languages helps students feel more comfortable, it can also reduce their exposure to essential design principles like abstraction and modularity. The paper argues that surface-level fluency should not replace the deeper analytical thinking that modern engineering roles demand.

Keywords: Industrial and Systems Engineering (ISE), Object-Oriented Programming (OOP), Engineering Education, Curriculum Development, Technical Competencies

ÖZET

Endüstri ve Sistem Mühendisliği (ESM), geleneksel olarak süreçlerin iyileştirilmesi ve kaynakların verimli kullanımı üzerine odaklanmaktadır. Ancak, hızla gelişen teknolojik ortamda, alan artık giderek daha fazla programlama, veri analizi ve algoritmik düşünme becerileri gerektirmektedir. Nesne Yönelimli Programlama (NYP) ve veritabanı bilgisi de dâhil olmak üzere programlama becerilerinin önemi artmış; modern endüstri ve sistem mühendisliği giderek daha fazla veri odaklı karar verme süreçlerine ve ileri düzey hesaplama araçlarına dayanmaya başlamıştır. Bu çalışma, programlama ve ilişkili teknik becerilerin ESM uzmanlarının eğitiminde ve mesleki faaliyetlerinde oynadığı rolü incelemektedir. Anket yanıtlarına dayanarak, mevcut ESM müfredatlarının sektör ihtiyaçlarını ne ölçüde karşıladığı ve geleneksel ESM bilgi birikiminin yanında teknolojik yetkinliklere daha güçlü bir vurgu gerekip gerekmediği ortaya konulmaktadır. Elde edilen bulgular, ayrıca doğrudan öğretim deneyimlerinden elde edilen içgörülerle de desteklenmektedir. Daha erişilebilir prototipleme dillerinin kullanılmasının öğrencilerin programlamaya karşı daha rahat hissetmelerini sağladığı görülürken, bunun soyutlama ve modülerlik gibi temel tasarım ilkelerine olan maruziyeti azaltabileceği de vurgulanmaktadır. Çalışma, yüzeysel bir akıcılığın, modern mühendislik rollerinin gerektirdiği derin analitik düşünmenin yerini almaması gerektiğini savunmaktadır.

Anahtar Kelimeler: Endüstri ve Sistem Mühendisliği(ESM), Nesne Yönelimli Programlama (NYP), Mühendislik Eğitimi, Müfredat Geliştirme, Teknik Yetkinlikler

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my thesis advisor, Prof. Ö. Erhun Kundakcıoğlu, for his invaluable guidance, support, and insightful feedback throughout every stage of this research. His expertise and encouragement were essential to the completion of this study.

I am also grateful to all participants who generously shared their time, experiences, and perspectives. Additionally, I gratefully acknowledge the support of the Scientific and Technological Research Council of Turkey (TUBITAK) under grant 122M839, which made this research possible.

Finally, I wish to express my heartfelt appreciation to my parents and friends for their unwavering encouragement, patience, and support throughout my academic journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Industrial and Systems Engineering (ISE), as a multidisciplinary field, has traditionally bridged the gap between engineering and management, offering solutions to optimize systems in manufacturing, logistics, technology, and operations. Historically, ISE has focused on process improvement, efficiency enhancement, and resource optimization. However, in the rapidly evolving technological landscape of the 21st century, the essence of what it means to be an Industrial and Systems Engineer is undergoing a significant transformation. The modern ISE professional is no longer confined to traditional production and operational paradigms but is increasingly required to engage with advanced computational tools, data analytics, and algorithmic thinking to remain relevant and effective. This evolution reflects a broader shift in industry and academia, raising questions about whether the integration of programming and technical skills into engineering education is becoming a core necessity. The competencies demanded of today's industrial engineers are expanding, with coding and algorithmic problem solving emerging as indispensable skills. Although the traditional ISE skillset remains critical, it is now increasingly supplemented by technical proficiencies, such as programming, automation, and digital modeling, which are becoming essential in navigating the modern, technology-driven landscape.

OR plays a central role in ISE, serving as a cornerstone for methodologies that tackle complex optimization problems. These problems often require advanced computational techniques to create effective solutions, creating a natural intersection between OR and programming. To gain proficiency in OR, one must not only understand mathematical models, but also be able to implement these models through programming and algorithmic design. In this context, programming is not just a supplementary skill, but a critical enabler for solving the intricate problems that OR addresses. To effectively model and implement solutions in the OR, proficiency in programming languages such as Python has become essential. Python, renowned for its simplicity and versatility, has emerged as the go-to tool for data optimization and manipulation across multiple industries, including manufacturing, logistics, and scientific research. Coding accelerates the development of efficient solutions to the complex problems addressed by OR, with languages such as Python, equipped with robust libraries for optimization and data manipulation, becoming integral to the process. High-profile organizations such as YouTube, Google, and NASA utilize Python for various applications, ranging from GIS mapping and scientific tasks to hardware testing [1]. Its general-purpose nature and widespread adoption across industries highlight the growing importance of programming skills, especially as Python continues to be a prominent feature in the job market, offering high demand and competitive salaries [1, 2]. This underscores the increasing relevance of programming expertise in the current landscape. Given the variety of programming languages designed for distinct purposes, understanding fundamental programming concepts and algorithmic thinking equips industrial and systems engineers with a broader perspective, even without being experts in specific languages.

This topic deserves further exploration, as some ISE students and professionals remain hesitant to fully embrace programming as a core competency. This reluctance often stems from the perception that coding is merely a supplementary skill rather than a fundamental requirement for career advancement. Many students still associate ISE primarily with managerial and organizational responsibilities, placing less emphasis on technical expertise. However, the evolving demands of modern industries challenge this perspective. As ISE expands into fields such as machine learning, data analytics, and smart manufacturing, programming is increasingly recognized as an essential tool for success. The debate persists, with some professionals and academics questioning the necessity of coding within the evolving framework of ISE.

Beyond survey insights, this study also draws on the practical teaching experience of the second author Ö. Erhun Kundakcıoğlu, who has observed that introducing more accessible, scripting-focused languages has made programming less intimidating for students. However, this approach sometimes comes at the cost of reducing exposure to fundamental principles like object-oriented thinking and system-level design. Striking the right balance between making coding approachable and ensuring that students develop deeper analytical and design skills remains an important challenge for curriculum development in ISE.

Significant variability exists in how programming and data-related skills are integrated into ISE curricula, even among accredited programs. For example, the University of Florida requires several mandatory programming and analytics courses, while Temple University and Virginia Tech offer similar content only as electives. Auburn University occupies a middle ground with at least one required course in programming and database applications [3, 4, 5, 6].

Moreover, some programs exclude database and programming topics altogether, while others provide only elective offerings. Even where present, dedicated OOP courses or substantial OOP content are often lacking. Basic Python scripting is increasingly common, but comprehensive database and OOP training remains limited in many curricula. Such variability suggests differences in how programs prioritize these technical skills, which may influence students' perceptions of their relevance and contribute to ongoing discussions about the core competencies for ISE professionals.

This study explores the evolving role of programming in ISE, emphasizing its integration into academic curricula and its practical applications across industries. Through an analysis of first-hand insights from industry professionals and academics gathered via a well-structured questionnaire, it seeks to bridge the gap between theoretical education and industry demands. By assessing the extent to which programming is embedded in modern ISE education and evaluating its impact across various sectors, this research aims to define the optimal balance between technical proficiency and managerial expertise for the industrial and systems engineer of the future.

The remainder of this study is structured as follows: Chapter 2 reviews relevant literature to provide context and background for the research. In Chapter 3, the research methodology is detailed, outlining the design of the survey, participant selection, and data collection procedures. Following this, Chapter 4 presents a thorough discussion of the research findings, including analysis and interpretation of the survey results in relation to the stated research questions. Finally, Chapter 5 provides a summary of the key insights gained from the study, highlights its limitations, and offers recommendations for future research and potential improvements in ISE education.

2. BACKGROUND

Industries across the globe are undergoing rapid digital transformation, fundamentally reshaping the skills required of ISE professionals. Traditional manufacturing expertise is no longer sufficient; today's professionals are increasingly expected to possess programming literacy, data analytics capabilities, and fluency in leveraging emerging technologies such as artificial intelligence. As a result, the once-clear boundaries between ISE, Operations Research (OR), and Computer Science (CS) are progressively dissolving, creating an urgent need for educational institutions to realign curricula with industry's evolving demands. This literature review explores how these shifting requirements have been addressed in scholarly research and identifies key directions for adapting ISE education to better prepare graduates for a dynamic and technologically driven professional landscape.

Technology and increasing complexity in industry have led to growing demands for technical skills among Industrial and Systems Engineers. Studies such as [7] indicate that competencies in programming, modeling, and statistics are frequently required in job postings. The expansion of ISE roles into information technology and service sectors further underscores this trend [8].

In line with these developments, [9] emphasize the importance of technical-technological competences in the context of Industry 4.0, highlighting media and coding skills as fundamental for ISE students and professionals. Additionally, since professionals face challenges when adopting Industry 4.0 and need to develop new skills or reinforce existing ones, the continuous improvement and redesign capabilities of ISE academics and practitioners should be leveraged to develop programs suitable for the digital era. The development of new business models, leading digital transformation efforts within organizations, and defining new roles for personnel in digitalized environments are among the value-added activities expected from ISE 4.0.

In parallel, educational institutions have responded to these evolving industry demands by revising and enhancing their curricula. Emphasis has been placed not only on strengthening programming skills but also on integrating emerging technologies into classroom settings. Faculty instructional strategies are being transformed to include more experiential learning opportunities, such as project-based assignments and industry collaborations, which better prepare students for real-world challenges. These comprehensive reforms aim to ensure that graduates possess the technical proficiency and adaptability required to thrive in a broadening range of ISE roles.

The increasing complexity of achieving comprehensive sustainable development is driven by the rapid advancements in technologies such as mass production, globalization, marketing, and digitalization. Consequently, higher education institutions are compelled to act as agents of change, necessitating their adaptation and reinvention across diverse contexts [10]. In response to these changes, recent research on the restructuring of ISE curricula has gained prominence, reflecting a notable shift in the educational landscape due to today's expectations. Modern Industrial and Systems Engineers typically use Predetermined Motion Time Systems, computer simulation (especially discrete event simulation), along with extensive mathematical tools and modeling and computational methods for system analysis, evaluation, and optimization [11].

These competencies, such as the application of advanced simulation techniques and mathematical models, align closely with OR methodologies. This convergence highlights the growing necessity of integrating OR skills within the ISE curriculum to meet the evolving demands of the profession. The incorporation of OR/CS methodologies not only addresses the growing demands of contemporary sectors but also highlights the necessity of these technical skills within the ISE curriculum. This connection serves as a key indicator of the broader transformations occurring within the field, underscoring the importance of adapting educational approaches to meet the evolving requirements of the industry. At this juncture, gaining a comprehensive understanding of the role of OR will be crucial for aligning educational practices with industry needs.

OR methodologies and models, as a branch of applied mathematics, play a critical role in various disciplines, including ISE. As a multidisciplinary applied science, OR integrates concepts from algebra and optimization, probability and statistics, and analyses in both real and abstract spaces. This integration has facilitated innovative approaches within ISE, encompassing machine learning and other advanced techniques [12]. Contemporary challenges in ISE are extensive and focus on enhancing productivity while addressing technical issues related to production management and control, as well as financial and economic impacts. OR methods are employed across diverse industrial settings, ranging from manufacturing and distribution to transportation, logistics, and services. The responsibilities of Industrial and Systems Engineers encompass a wide range, from designing individual operations to managing and controlling entire production, logistics, and service systems [12].

Given the multifaceted perspective of OR, it is clear that competencies beyond mathematical proficiency are increasingly valuable. Although OR and CS might initially seem such as distinct disciplines, their significant overlap is crucial. In this context, the work of the ICS Education Committee, established by the INFORMS Computing Society in 2007, is particularly noteworthy. The committee was tasked with developing a curriculum to prepare students for graduate studies and industrial roles at the intersection of OR and CS disciplines. Their report highlights the shared needs of both fields and proposes an undergraduate curriculum designed to bridge the OR/CS interface, thereby preparing students for both professional roles and advanced academic paths [13]. The report emphasizes that students pursuing careers in OR/CS disciplines, whether in industry or academia, must possess proficiency in modern OOP languages (such as C++,C#, Java) and one or more scripting languages. This proficiency is essential due to the widespread use of these development platforms in contemporary computing environments. The report also highlights the necessity of being able to write concurrent or parallel programs to address runtime issues, as many significant problems have algorithms with exponential time complexities. Furthermore, it underscores that large-scale OR problems require extensive computational solutions involving numerous modules and complex interdependent data relationships, making expertise in software engineering and system analysis indispensable for large-scale software development endeavors [13].

Similarly, in their study, [7] analyzed job advertisements to identify the skills that employers expect from professionals in OR. They highlighted eleven key skills, which include both soft and technical competencies. Modeling emerged as the third most important skill, with a prevalence of 52%. This category encompasses not only optimization and mathematical programming but also forecasting, model development, and simulation. Statistics followed closely in fourth place at 48%, with requirements ranging from basic data analysis to specific statistical techniques, such as logistic regression. This category also includes dataset manipulation and data mining. Additionally, programming skills ranked fifth at 47%, while general analytical ability was noted at sixth with a 42% prevalence. The study emphasizes the challenges of imparting all these soft and technical skills within an undergraduate education, while making a strong case for the inclusion of modeling, statistics, programming, and general analytical skills in an OR-specific curriculum.

The findings of [13] and [7] reveal parallel insights regarding the essential skills required in the fields of OR and CS. Both studies underscore the increasing demand for competencies in programming, modeling, and statistics, highlighting their significance in preparing students for professional roles in these disciplines. This alignment emphasizes the necessity for educational curricula to evolve, ensuring that students acquire the relevant skills to meet industry expectations effectively. At this point, the study by [14] offers a broad and multidimensional perspective on the issue. They argue that addressing the challenges faced by the OR/MS community requires a comprehensive approach. This means viewing the OR/MS community as a ”business ecosystem,” where educators, practitioners, and researchers form the core. The sustainability and growth of this ecosystem depend on the support of key stakeholders, including senior and middle management, engineering leaders from both the private and public sectors, professional societies, universities, and research funding agencies. These stakeholders play an essential role in maintaining the core’s strength. Without robust collaboration among these communities to meet the needs of end users, the OR/MS ecosystem risks becoming marginal and confined to niche areas in the future.

Beyond these structural and community-level considerations, technological innovations are also redefining how knowledge is acquired, shared, and applied within OR, IE, and related disciplines. Among these innovations, the rise of artificial intelligence (AI) tools introduces new dynamics to the conversation. For example, large language models (LLMs), based on Natural Language Processing (NLP) techniques, have demonstrated the capacity to support personalized learning by generating tailored responses to student queries and feedback [15, 16]. These AI-driven tools facilitate academic success by adapting content and learning strategies to individual needs, enabling real-time assistance regardless of location [16]. However, despite their benefits in enhancing learning efficiency and communication, the application of such AI tools also raises concerns and potential risks, especially in technical fields like computer programming [17, 18]. These challenges underscore the importance of balancing AI integration with the development of critical thinking and problem-solving skills.

[19] points out that while ChatGPT shows promise as a conversational agent, it also faces notable limitations. The model struggles with complex calculations, exhibits bias in generating preferred digits, and can give contradictory or overconfident responses

to minor question changes. [18] discuss potential threats in their study under several headings. The first of these is that overreliance on generative AI tools can negatively impact education and research by diminishing critical thinking and problem-solving skills due to the ease of accessing ready-made answers and solutions. Another significant aspect discussed is the impact on critical thinking and problem-solving skills. AI tools demonstrate the ability to generate nearly accurate answers to a broad range of technical questions and can also correct or partially amend programming code based on problem descriptions, algorithms, and problem names. Nevertheless, depending on ChatGPT for answers and code may hinder learners' development of their critical thinking and problem-solving capabilities. Lack of creativity, lack of contextual understanding, and limited comprehension are additional potential risks identified in the literature [20].

Therefore, AI language models (LLMs) are valuable tools for education and research. As a groundbreaking LLM, ChatGPT can engage in human-like conversations and produce text that closely mimics human writing. Its capabilities include answering questions, composing essays, solving problems, explaining complex topics, providing virtual tutoring, practicing languages, and assisting in programming education. Furthermore, ChatGPT is equipped to tackle both technical (e.g., engineering and computer programming) and non-technical (e.g., language and literature) challenges [18]. It is crucial to recognize that solutions generated by AI tools may not always be precise during their initial attempts, which requires students to engage in critical evaluation of the responses they receive. They have been noted to sometimes produce inaccurate information concerning people, places, or facts. As a result, students need to draw upon their subject knowledge to make informed assessments. This critical evaluation process can significantly enrich their learning experience. Furthermore, as ChatGPT evolves and learns from its inaccuracies, its performance is likely to improve over time, providing further benefits to students [21].

At this point, considering the perspective and experience of professionals will also provide a broader outlook on the topic. When considering the experiences of professionals, we also observe that those with a certain level of expertise often use these tools to seek assistance. Additionally, similar to comments made in academic discussions, there are concerns that these tools may harm creativity. As highlighted in the study by [22], although AI tools can be useful for learning or automating routine tasks, they can, in fact, reduce productivity in some areas. Its use can diminish team communication, as questions that might be better addressed to a colleague are sometimes directed to the AI, which in turn reduces overall collaboration. Moreover, an overconfident AI-generated response may give participants the false impression that further discussion with the team is unnecessary, and participants may spend excessive time refining prompts to generate perfect outputs, rather than quickly resolving small defects on their own [22].

In summary, the integration of programming and analytical skills into the curriculum of ISE and OR is a critical step not only in responding to the evolving demands of modern industries but also in equipping students to navigate the complexities of their future roles. Throughout this literature review, a consensus has emerged among researchers and industry professionals regarding the necessity of empowering students with competencies in programming, modeling, and statistics. The increasing interdependence between OR and CS further underscores the importance of a multidisciplinary approach to education. While AI tools offer significant support in coding and analytics, foundational programming skills remain crucial for effectively leveraging these technologies.

3. METHODOLOGY

3.1 Conceptual Framework and Research Design

In this study, the ISE curriculum is examined against evolving expectations in industry and academia, with particular attention to whether programming, especially OOP, and database skills should be treated as core outcomes. As digital systems permeate engineering work, integrating programming competence into the ISE toolkit has become essential for addressing complex problems and delivering implementable solutions.

The analysis considers both the general need for programming and database knowledge in ISE and the specific OOP competencies required in professional practice. Mastery of core programming and database concepts, together with OOP principles, enables ISE professionals to design efficient solutions, build scalable applications, and manage complex systems. The study also takes into account broader capacities—programming literacy (the ability to understand and use programming concepts) and technological adaptability (the ability to learn and apply new tools quickly)—reflecting the multifaceted skill set expected of today's ISE professionals.

In this context, it is important to distinguish between general programming (coding) and OOP. While programming broadly means writing instructions for computers, OOP is a paradigm that organizes code into objects and classes, promoting modularity and reusability. This approach matches well with the ISE mindset, which values systems thinking and modular design to solve complex problems effectively. Therefore, OOP skills support industrial and systems engineers in developing flexible and maintainable solutions.

In addition, the ACID principles—Atomicity, Consistency, Isolation, and Durability—are fundamental to database systems and ensure that data transactions are processed reliably and without error, which is critical when managing complex industrial operations.

In today's professional landscape, regardless of sector or department, all professionals increasingly need to work with data effectively. Therefore, beyond the inherent benefits of these principles, understanding core database concepts helps engineers know how to properly manage, utilize, and interpret data—an essential skill in data-driven decision-making and process improvement.

Similarly, certain SOLID principles in OOP, especially Single Responsibility, Open-Closed, and Dependency Inversion align closely with the ISE mindset. The Single Responsibility Principle encourages clear separation of tasks, mirroring how ISE emphasizes dividing complex processes into manageable parts. The Open-Closed Principle supports designing systems that can evolve without major overhauls, which is critical in dynamic industrial environments. Dependency Inversion fosters modular and loosely coupled components, facilitating flexible and maintainable process designs. By focusing on these key principles, ISE students cultivate the ability to develop adaptable, efficient solutions applicable across diverse professional fields.

Through the application of these principles, ISE students develop a mindset that facilitates process improvement and effective management. This mindset applies across diverse professional contexts. These proficiencies are therefore essential regardless of the specific professional domain, justifying a detailed examination of programming, database knowledge, and their underlying principles in this study. Furthermore, this methodological approach, which encourages analytical thinking rather than rote memorization, provides industrial and systems engineers with a distinctive perspective for professional growth, equipping them with abilities that enhance their problem-solving and process optimization skills in complex, real-world environments.

The conceptual framework outlined above provides the foundation for the specific research questions explored in this study.

3.2 Research Questions

To address these objectives, four key research questions are explored to examine the role of programming, particularly OOP, within the ISE field and its impact on professional development as well as the current curriculum. These research questions are as follows:

1. To what extent do engineering professionals consider programming and database knowledge, particularly OOP, as essential for their roles in modern industry? How would this differ based on technology development in their department (sales and marketing, R&D, IT, etc.) and sector (FMCG, Tech Start-up, Finance, Healthcare, etc.)?

2. How do engineers perceive the adequacy of their university education in preparing them with the necessary programming and database skills, to address contemporary challenges in their fields? How would this differ based on country and technology development in their department (sales and marketing, R&D, IT, etc.) and sector (FMCG, Tech Start-up, Finance, Healthcare, etc.)?

3. What is the relationship between OOP and database proficiency and the professional development, project success, and problem-solving capabilities of engineers?

4. How do AI-based tools, such as ChatGPT, influence the daily work and decision-making processes of engineers, and what role does programming knowledge play in effectively utilizing these tools? How would this differ based on their department (marketing, supply chain, etc.)?

3.3 Data Collection and Survey Instrument

To gather relevant insights, a comprehensive 30-question survey was developed using SurveyMonkey and distributed via email and other digital channels to a diverse group of ISE professionals, including individuals from academia, industry, and those active in both domains. Participants represented a variety of sectors such as technology, finance, and academia. The survey aimed to explore the role of programming—particularly OOP and database knowledge, and the use of AI-based tools in professional settings. These insights are essential for understanding the current state of technical proficiency in the ISE field and for evaluating whether existing curricula sufficiently prepare graduates for real-world demands.

The questionnaire was structured to align with the four research questions. It examined the perceived importance of programming and database knowledge, the adequacy of university education in preparing engineers for technical challenges, the role of programming proficiency in shaping career outcomes such as problem-solving and advancement, and the extent to which AI tools influence daily engineering tasks.

By aligning the survey questions with the research objectives, this study aims to provide a clear connection between industry needs, academic training, and the role of programming in shaping the careers of ISE professionals. The findings from the survey will help identify potential gaps in the curriculum and inform recommendations for the integration of more relevant programming competencies into ISE education.

3.4 Data Analysis

To explore the research questions outlined in this study, a series of survey questions were designed to directly relate to each of the key areas of inquiry. Below is a summary of how the survey questions align with each research question:

3.4.1 Research Question 1

To what extent do engineering professionals consider programming and database knowledge, particularly OOP, as essential for their roles in modern industry? How would this differ based on technology development in their department (sales and marketing, R&D, IT, etc.) and sector (FMCG, Tech Start-up, Finance, Healthcare, etc.)?

Related Survey Questions:

- Question 1: How would you rate your programming knowledge? (This question helps assess the level of programming knowledge that professionals have and is key to understanding if they feel it's essential for their roles.)
- Question 5: How critical is programming knowledge for your role? (Directly addresses the importance of programming skills in the professional roles of engineers.)
- Question 7: How would you rate your knowledge of OOP concepts? (Specifically focuses on OOP, which is a key part of this research question.)
- Question 10: Have you encountered situations where the lack of proficiency in OOP concepts limited your effectiveness? (Explores the impact of lacking OOP skills on professional effectiveness, tying directly to this research question.)
- Question 12: How would you rate your knowledge of Database concepts? (This question helps assess the level of database knowledge that professionals have and is key to understanding if they feel it's essential for their roles.)
- Question 15: Have you encountered situations where the lack of proficiency in database concepts limited your effectiveness? (Explores the impact of lacking database skills on professional effectiveness, tying directly to this research question.)

3.4.2 Research Question 2

How do engineers perceive the adequacy of their university education in preparing them with the necessary programming and database skills, to address contemporary challenges in their fields? How would this differ based on country and technology development in their department (sales and marketing, R&D, IT, etc.) and sector (FMCG, Tech Start-up, Finance, Healthcare, etc.)?

Related Survey Questions:

- Question 6: How adequate was the programming knowledge you gained in university for your professional life? (This question directly asks about the adequacy of university education in preparing for the professional application of programming skills.)

- Question 9: Was your university education sufficient for applying OOP in real-world industry problems? (Specifically targets OOP, which is central to this research question.)

- Question 14: Do you feel that your university education sufficiently covered Database concepts and their applications? (While this focuses on databases, it touches on the broader theme of how well educational programs cover technical competencies.)

3.4.3 Research Question 3

What is the relationship between OOP and database proficiency and the professional development, project success, and problem-solving capabilities of engineers?

Related Survey Questions:

- Question 10: Have you encountered situations where the lack of proficiency in OOP concepts limited your effectiveness? (This question helps link the lack of technical skills to job performance, similarly to how proficiency in OOP might impact an engineer's career.)

- Question 11: Which aspects of your career have been positively influenced by proficiency in OOP concepts? (Directly addresses the relationship between OOP proficiency and career-related factors such as promotion, project success, and problem-solving.)

- Question 15: Have you encountered situations where the lack of proficiency in database concepts limited your effectiveness? (This question helps link the lack of technical skills to job performance, similarly to how proficiency in database concepts might impact an engineer's career.)

- Question 16: Which aspects of your career have been positively influenced by proficiency in Database concepts? (Though focused on databases, this question is useful to assess the broader connection between technical proficiency and career outcomes.)

3.4.4 Research Question 4

How do AI-based tools, such as ChatGPT, influence the daily work and decision-making processes of engineers, and what role does programming knowledge play in effectively utilizing these tools? How would this differ based on their department (marketing, supply chain, etc.)?

Related Survey Questions:

- Question 17: How often do you use ChatGPT or similar AI-based tools? (Directly addresses the frequency of AI tool usage, which is central to this research question.)

- Question 18: For what purposes do you use ChatGPT or similar AI tools? (Examines the specific use cases of AI tools and how they relate to the work of engineers.)

- Question 19: How much do you trust the outputs provided by ChatGPT or similar AI-based tools? (This question helps gauge engineers' confidence in using AI tools, which connects to their perception of how programming knowledge impacts the efficacy of these tools.)

- Question 21: How significant do you believe prior knowledge in programming or databases is for effectively utilizing ChatGPT or similar AI tools? (Directly ties programming knowledge to the effective use of AI-based tools, addressing the core of this research question.)

By structuring the questionnaire around these specific research questions, the study aims to draw clear connections between the industry needs, academic preparation, and the evolving role of programming knowledge in the professional development of industrial and systems engineers. While these connections were established before data collection, the deeper insights and analysis regarding the professional backgrounds and industry-specific trends will be provided in the findings section.

3.5 Data Processing and Visualization

The survey data collected was processed and analyzed using Python programming language. Key libraries such as pandas, numpy, matplotlib, and seaborn were employed to clean, organize, and visualize the data. These tools enabled the extraction of meaningful insights regarding the participants' programming and database proficiency, as well as their perceptions of AI tool usage.

The data analysis involved:

- Importing raw survey data into pandas DataFrames for efficient manipulation.
- Cleaning and filtering data based on relevant demographic and professional attributes (e.g., sector, department, country).
- Generating visualizations such as bar charts, pie charts, and box plots to illustrate distributions and trends across different groups.

This approach facilitated an objective and reproducible analysis process, ensuring transparency and robustness in the study's findings.

4. RESEARCH FINDINGS AND DISCUSSION

This chapter presents the main empirical results of the study by combining descriptive and analytical perspectives. It is divided into two parts: the first offers an overview of the distributions and trends observed in individual survey items, while the second provides a focused analysis structured around the four research questions.

4.1 Overview of Survey Findings

This section offers a descriptive overview of participants' responses to individual survey items. It highlights general trends and distributions across key variables, laying the groundwork for the subsequent research question-driven analyses.

A total of 104 participants completed the survey. The majority of respondents hold an undergraduate degree in ISE, aligning with the focus of this study. However, contributions from individuals with backgrounds in various STEM disciplines such as CS, Electrical Engineering, Mathematics, and Mechanical Engineering were also incorporated, offering a broader perspective. While a significant portion of the participants currently work in academic or research positions, many are also actively involved in the industry as consultants or technical experts.

Most participants completed their undergraduate studies between 2011 and 2020, with Turkey and the United States being the most frequently reported countries of undergraduate education and current employment. In terms of professional experience, the sample includes individuals across various career stages, with a notable concentration in the 4 to 14 years range. Participants occupy diverse positions such as managers, academics, engineers, software developers, and data professionals. Departmental affiliations primarily include IT, production and planning, R&D, and finance. A more detailed breakdown of demographic characteristics is provided in Appendix A, where graphs corresponding to the first nine questions which is Professional Background section of the survey are included.

The second part of the survey, titled Programming Knowledge and Experience, included six questions designed to assess participants' programming proficiency and habits. The analysis of this section offers insights into respondents' perceived skill levels, commonly used programming languages, learning resources, and the professional application of coding.

As illustrated in Figure 4.1, the majority of participants evaluated their programming skills as intermediate or advanced. Only a small portion of respondents identified themselves as novices.

Figure 4.1: *Distribution of Self-Assessed Programming Proficiency Levels*

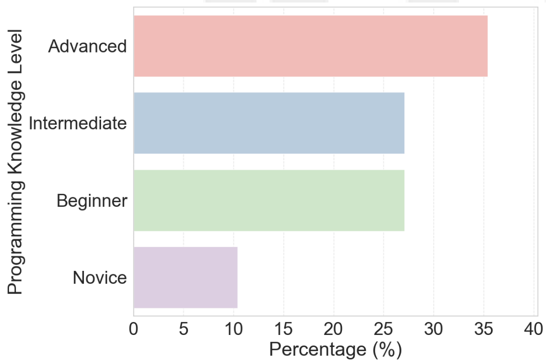


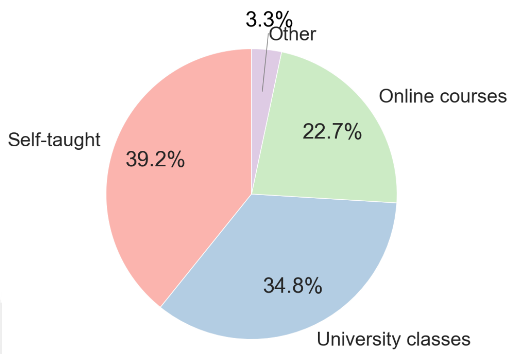
Figure 4.2: *Programming Languages Used by Respondents*

Language	# of Respondents
SQL	59
Python	56
Java	23
C++	19
C	12
C#	10
JavaScript	8
Matlab	5
R	4
Bash	1
KQL	1
Scala	1
None	1
Julia	1

Figure 4.2 highlights that SQL and Python emerge as the most widely used languages for programming and querying among the respondents, with 59 and 56 mentions respectively. Java and C++ follow with moderate levels of usage. Meanwhile, languages such as Bash, KQL, Scala, and Julia are each mentioned only once, indicating minimal adoption within the participant group.

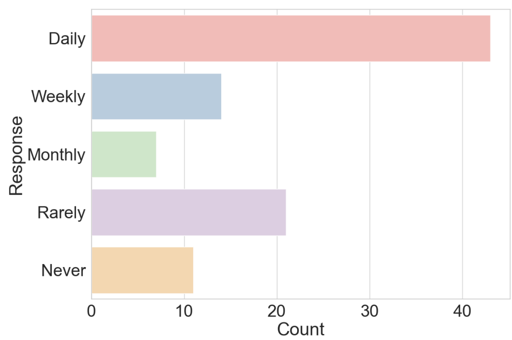
While the distribution of respondents' sources for learning programming, as shown in Figure 4.3, is noteworthy and indicates that most participants are self-taught, it also demonstrates that university education plays a supportive role in their learning process. However, this raises the need for a deeper analysis regarding whether university education alone is sufficient. The data suggests that respondents tend to rely on additional resources or methods alongside their university education.

Figure 4.3: *Programming Knowledge Acquisition Methods Among Respondents*



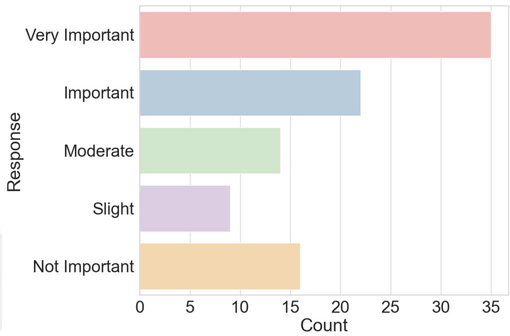
The frequency of programming language usage among respondents, as illustrated in Figure 4.4, reveals that a significant portion (43 participants) use programming daily in their work. Weekly and rarely users follow, with 14 and 21 respondents respectively, while a smaller group reports monthly or no usage at all. This distribution suggests that programming is an integral part of the daily workflow for many, yet a considerable number use it less frequently, indicating varied reliance on programming skills across the sample.

Figure 4.4: *Frequency of Programming Language Usage Among Respondents*



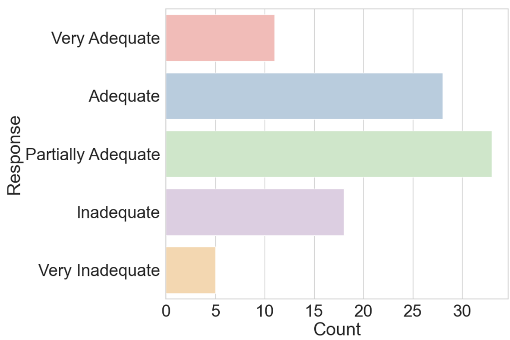
As presented in Figure 4.5, a substantial proportion of respondents perceive programming knowledge as either very important (35) or important (22) for their professional roles. Although a smaller segment considers it moderate (14) or slight (9) in importance, a notable minority (16) view it as not important. This distribution highlights the diverse range of programming dependency across different roles within the sample.

Figure 4.5: *Respondents' Perception of Programming Relevance*



The responses depicted in Figure 4.6 suggest that while a portion of participants found their university programming education to be either ”Adequate” or ”Very Adequate”, the majority rated it as only ”Partially Adequate”. Notably, 23 respondents (combining ”Inadequate” and ”Very Inadequate”) expressed that their education fell short in preparing them for professional programming tasks. This highlights a potential gap between academic instruction and real-world programming demands.

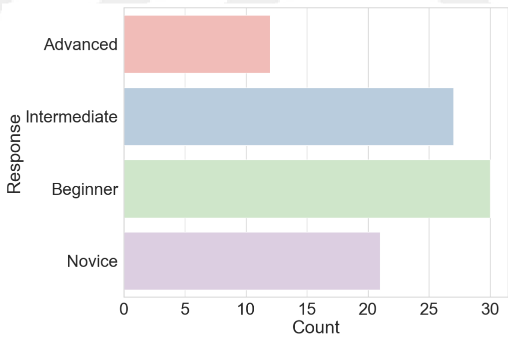
Figure 4.6: *Adequacy of University Programming Education for Professional Needs*



The third section of the survey investigates participants' familiarity with and perceptions of OOP concepts. It comprises five questions designed to explore the respondents' self-assessed knowledge levels, awareness of core OOP principles such as SOLID, the adequacy of university training, and the perceived professional impact of OOP proficiency.

The self-assessed proficiency in OOP concepts is depicted in Figure 4.7. While a considerable portion of respondents rated themselves as beginners (30) or intermediate (27), a relatively smaller group identified as advanced (12). This distribution suggests that although OOP is a foundational topic, a large segment of professionals still feel they have room for improvement in this area. Notably, 21 respondents categorized their understanding as novice, which may indicate potential areas for further development depending on individual roles and professional needs.

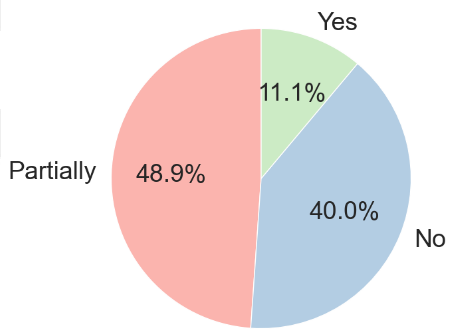
Figure 4.7: *Participants' OOP Knowledge Level*



This generally limited proficiency is also evident in participants' familiarity with more advanced concepts. For instance, only 20% of respondents reported awareness of the SOLID principles of OOP, indicating that fundamental software design principles are not widely known among ISE graduates. This observation underscores a broader lack of exposure to advanced OOP practices, which may stem from varying role demands or indicate areas where additional educational support could be considered.

A potential contributing factor to this lack of familiarity may be the limitations of formal education in covering practical and advanced aspects of OOP. Figure 4.8 presents respondents' perceptions regarding the adequacy of university education in preparing them for applying OOP in real-world industry problems. Nearly half (48.9%) stated that their education was only partially sufficient, while 40% found it entirely insufficient. Only 11.1% felt adequately prepared. This data underscores a recurring theme across the study: although OOP is taught in academic settings, many graduates struggle to translate theoretical knowledge into practical applications in the workplace.

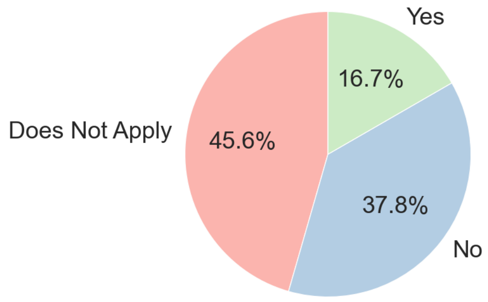
Figure 4.8: *Perceived Adequacy of University Education for Real-world OOP Application*



While the previous questions explored participants' educational background and self-assessed proficiency in OOP, the following items aim to examine how these knowledge levels are reflected in their professional experience.

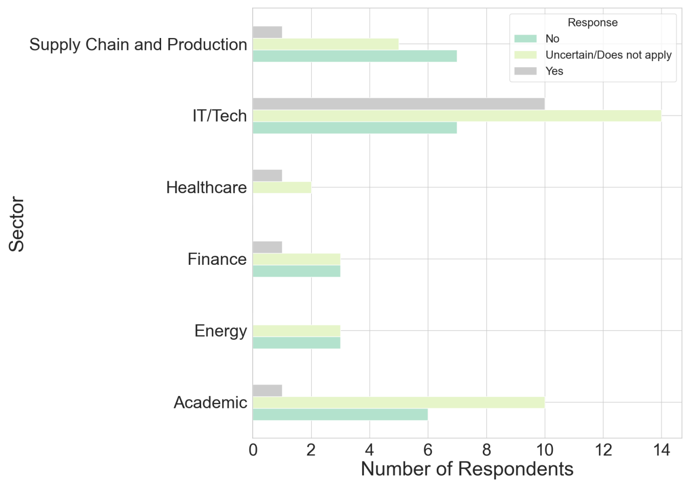
Figure 4.9 illustrates that 16.7% respondents acknowledged that a lack of OOP proficiency had negatively impacted their professional effectiveness, while 37.8% explicitly stated that it had not. A notable portion 45.6% chose "Uncertain/Does not apply," suggesting that the effect of OOP proficiency on job performance may vary significantly depending on individual roles and responsibilities.

Figure 4.9: *Impact of Lack of OOP Proficiency on Professional Life*



To better understand whether the reported limitations due to lack of OOP proficiency vary by professional context, it is insightful to cross-reference these responses with participants' sectors. Figure 4.10 displays this relationship, potentially highlighting whether certain fields are more sensitive to OOP proficiency than others. To ensure the reliability of the visualized trends, sectors with fewer than three responses were excluded from the analysis.

Figure 4.10: *Sector-wise Breakdown of OOP Proficiency-Related Challenges*



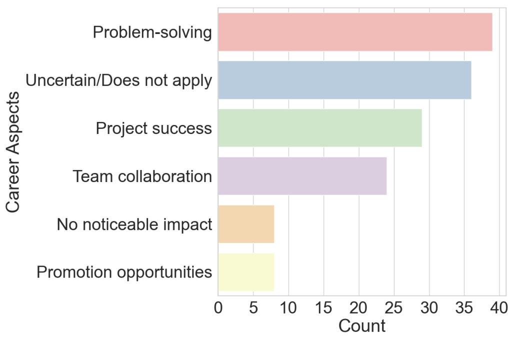
The results reveal variation in perceived limitations due to lack of OOP proficiency across sectors. While the IT/Tech sector shows a considerable number of "Yes" responses

(10), indicating notable constraints when OOP skills are lacking, it also has a substantial portion of “Uncertain/Does not apply” responses (14). This suggests that even within technically oriented fields, the relevance or impact of OOP proficiency can vary depending on specific roles or contexts. Sectors like Academic and Production predominantly report “No” or “Uncertain/Does not apply,” implying that OOP proficiency limitations are either less critical or not directly applicable in these areas.

In summary, while the lack of OOP skills presents significant challenges in certain technical sectors, the overall impact is nuanced and depends on the particular nature of work within each sector.

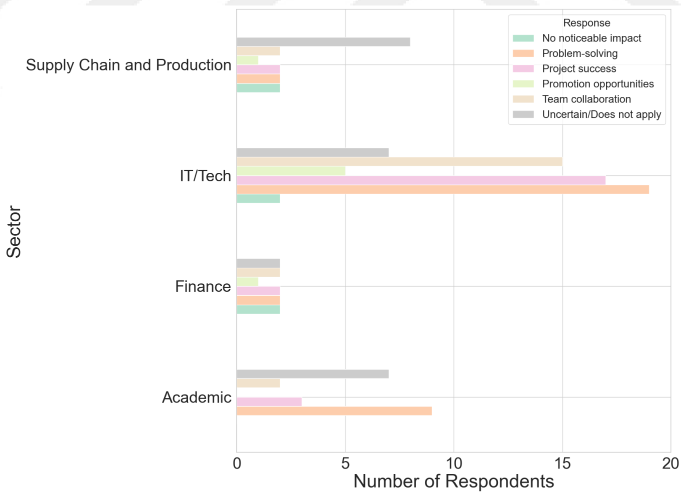
On the other hand, Figure 4.11 illustrates the perceived impact of OOP proficiency on various aspects of participants’ professional careers. Among the listed outcomes, problem-solving emerged as the most frequently cited benefit, selected by 39 respondents. This was followed by project success (29 respondents) and team collaboration (24 respondents), suggesting that strong OOP skills may enhance both individual performance and collaborative efficiency in software-related tasks. Notably, only a small number of participants reported promotion opportunities (8) or no noticeable impact (8), while a relatively high number (36) selected “Uncertain/Does not apply.” This indicates that the influence of OOP on career development may be highly role-specific or not always explicitly recognized by professionals. These findings reinforce the idea that although OOP proficiency contributes positively to technical and team-based outcomes, its broader career implications may remain indirect or underacknowledged in certain work environments.

Figure 4.11: *Perceived Positive Career Outcomes from OOP Knowledge*



Compared to the previous question on limitations arising from insufficient OOP knowledge, this item offers a more positive outlook by highlighting areas where OOP proficiency serves as an asset. To gain a more nuanced understanding of these outcomes, Figure 4.12 breaks down the reported positive impacts of OOP proficiency by sector.

Figure 4.12: *Perceived Positive Career Outcomes from OOP Knowledge by Sector*



The analysis in Figure 4.12 presents the distribution of perceived positive impacts of OOP proficiency across different professional sectors. To enhance clarity and focus on meaningful trends, the data was filtered to include only sectors with at least 5 total positive responses and positive impact categories that received 10 or more total mentions across all sectors. This approach excludes less-represented sectors and impact types, reducing noise from sparse data.

As a result, seven sectors remain in the analysis: Academic, Chemical, Finance, Healthcare, IT/Tech, Oil and Gas, and Production. Among these, the IT/Tech sector clearly dominates across nearly all positive impact categories, reflecting its strong reliance on OOP skills. For instance, IT/Tech professionals reported the highest numbers in Problem-solving (18), Project success (17), and Team collaboration (15), suggesting that OOP proficiency contributes significantly to both individual capabilities and collaborative work in this sector. The Academic sector also stands out, especially in Problem-solving (9) and Uncertain/Does not apply (7) categories, indicating a mixed perception where some find OOP knowledge highly beneficial while others are uncertain about its direct career impact. Other sectors such as Finance, Production, Chemical, Healthcare, and Oil and Gas show moderate but consistent positive responses, predominantly in Problem-solving and Project success, suggesting that OOP skills play a supportive role in enhancing work outcomes even outside traditional software-focused domains.

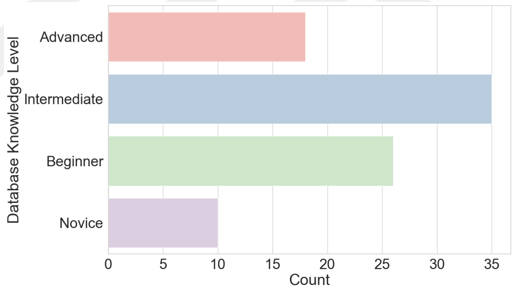
Therefore, the filtered analysis emphasizes that the practical benefits of OOP proficiency are most pronounced in sectors with direct software development or technical problem-solving roles, while sectors with fewer responses or less direct reliance on programming skills report lower impact levels.

Overall, while OOP proficiency appears to enhance performance in technically oriented sectors, its broader career impact remains role- and context-dependent, with limited influence in domains where OOP is not a central part of day-to-day responsibilities.

The fourth section of the survey explores participants' knowledge and experience related to database concepts. It comprises five questions aimed at assessing respondents' self-evaluated understanding of fundamental database principles, their awareness of ACID (Atomicity, Consistency, Isolation, Durability) properties, and the perceived adequacy of their university education on database-related topics. Additionally, the section investigates whether a lack of proficiency in database concepts has ever hindered professional performance, and identifies which aspects of respondents' careers have been positively influenced by such proficiency.

As illustrated in Figure 4.12, participants were asked to evaluate their level of knowledge regarding fundamental database concepts. The responses indicate a predominantly intermediate level of self-assessed competence. Specifically, 35 respondents categorized their knowledge as Intermediate, while 18 considered themselves Advanced. On the other hand, 26 participants identified as Beginners, and 10 selected Novice. This distribution suggests that while a substantial portion of the sample possesses moderate familiarity with database concepts, there remains a significant cohort with only foundational or limited understanding.

Figure 4.13: *Self-assessed Knowledge Levels Regarding Database Concepts*

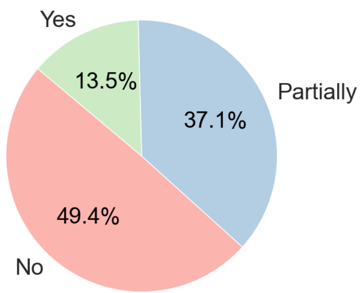


To assess the conceptual depth of participants' database knowledge, the survey included a question on familiarity with ACID (Atomicity, Consistency, Isolation, Durability) principles. A substantial majority (76.4%) indicated unfamiliarity with these concepts, revealing a gap in theoretical exposure despite many having self-assessed their knowledge as intermediate or higher.

While the first two questions focused on participants' self-perceived knowledge and conceptual awareness, the following items aim to uncover how formal education has contributed to this knowledge base, the practical consequences of limited database proficiency, and the specific professional domains where such knowledge proves beneficial.

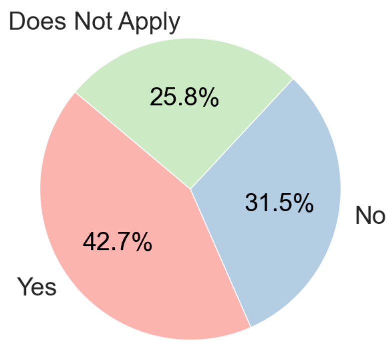
As shown in Figure 4.14, when asked whether their university education sufficiently covered database concepts and their applications, a significant portion of respondents expressed dissatisfaction. Nearly half of the participants (49.4%) indicated that their education did not adequately address these topics, while only 13.5% felt that their education was sufficient. The remaining 37.1% provided a more moderate response, stating that database education was only partially adequate. This distribution highlights a potential gap between formal education and the practical demands of professional environments, particularly in fields where database management skills are essential.

Figure 4.14: *Perceptions of University Education Coverage on Database Concepts*



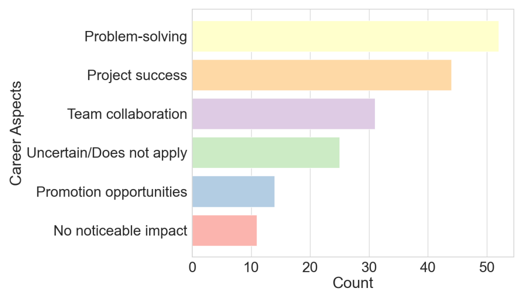
To assess the practical consequences of database-related knowledge gaps, participants were asked whether they had experienced professional limitations due to insufficient proficiency in database concepts. As depicted in Figure 4.15, approximately 43% of the respondents confirmed that such limitations had indeed impacted their effectiveness. Meanwhile, around 32% reported no such experiences, and 26% were either uncertain or felt the question did not apply to their context. These findings indicate that, for a significant portion of professionals, inadequate knowledge of database principles may translate into tangible challenges in the workplace.

Figure 4.15: *Impact of Database Proficiency Gaps on Professional Effectiveness*



The final item investigated the perceived career-related benefits associated with database proficiency. As respondents were allowed to select multiple options, the percentages exceed 100%. The most frequently reported outcomes were enhanced problem-solving skills (58.4%) and increased project success (49.4%), underscoring the technical and analytical value of database competence. Team collaboration (34.8%) and promotion opportunities (15.7%) were also noted, though to a lesser extent. A subset of respondents reported no noticeable impact (12.4%) or remained uncertain / not applicable (28.1%).

Figure 4.16: *Positive Impacts of Database Proficiency on Professional Effectiveness*



These findings suggest that, while not universal, database proficiency is perceived to positively influence a range of professional domains, particularly those requiring structured analysis and cross-functional collaboration.

Overall, the survey results highlight varying levels of database knowledge and awareness among participants, with a majority reporting intermediate proficiency but limited familiarity with foundational concepts like ACID. Formal university education appears insufficient for many, which correlates with reported limitations in professional effectiveness due to gaps in database skills. Nevertheless, proficiency in database concepts is recognized as beneficial across multiple career aspects, especially in problem-solving and project success. These insights collectively emphasize the importance of strengthening database education to better prepare professionals for practical challenges.

In the fifth and final section of the survey, five questions were included to examine participants' engagement with AI-based tools. This section focused on the frequency and purpose of usage, the level of trust and satisfaction with these tools, and perceptions regarding the relevance of prior technical knowledge—particularly in programming and database concepts—for effective utilization. The motivation behind this part was to explore whether the increasing use of AI tools can compensate for the limited exposure to foundational computing education often observed in undergraduate programs, or whether such tools are only truly effective when built upon an existing base of technical knowledge.

The survey results regarding the frequency and purposes of AI-based tool usage, as illustrated in Figures Figure 4.17 and Figure 4.18, reveal important trends about participants' engagement with these technologies.

Figure 4.17 shows that nearly half of the respondents (47.2%) report using AI tools on a daily basis, while an additional 28.1% engage with them weekly. Conversely, a small minority indicated rare or no usage. This pattern suggests that AI tools have become integral to the routines of many users, highlighting their growing role in supporting academic and professional tasks.

Figure 4.18 further explores the motivations behind AI tool usage. The majority of respondents utilize these tools primarily for learning and research purposes (35.3%), followed by programming assistance (23.7%) and problem solving (17.4%). Uses such as content creation and other miscellaneous purposes were less common. This distribution underscores the instrumental role AI tools play in supplementing technical skills and knowledge, especially in areas where formal education may fall short.

Figure 4.17: *Frequency of AI Tool Usage Among Participants*

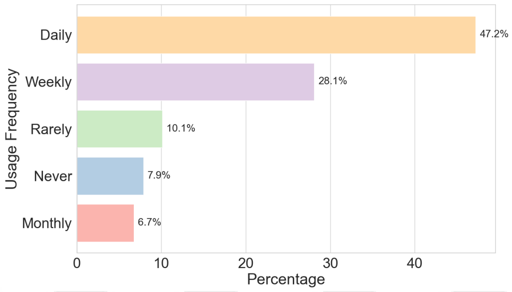
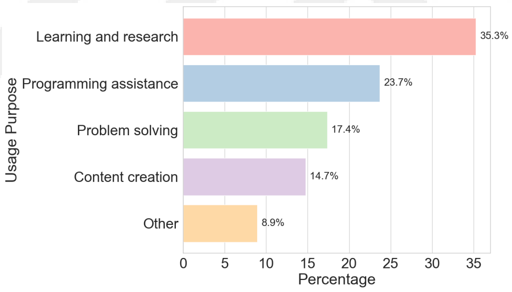


Figure 4.18: *Participants' Primary Purposes for Using AI-Based Tools*



Subsequently to evaluate users' overall perceptions of AI-based tools, two consecutive questions in the survey focused respectively on trust in AI outputs and satisfaction with usage experience. As shown in Figure 4.19, more than half of the respondents (51.69%) identified as Neutral regarding their trust in AI outputs, followed by 37.08% who expressed Trust, and only a small minority (3.37%) who indicated Complete Trust. In contrast, Figure 4.20 demonstrates a more favorable trend in terms of user satisfaction: a combined 80.9% of participants reported being either Satisfying (59.55%) or Very Satisfying (21.35%) with their experience. This disparity suggests that while users may hesitate to fully trust AI-generated outputs, they still perceive these tools as beneficial and effective in practice.

Figure 4.19: *Participants' Trust in AI Tool Outputs*

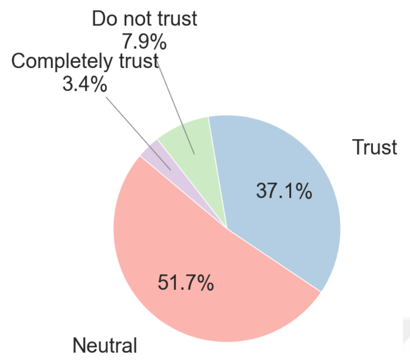
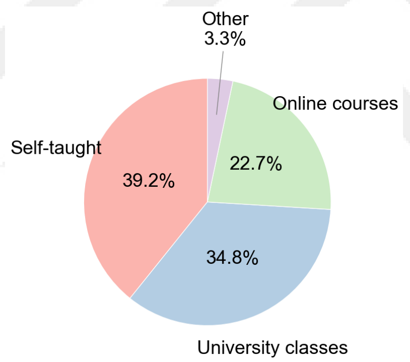
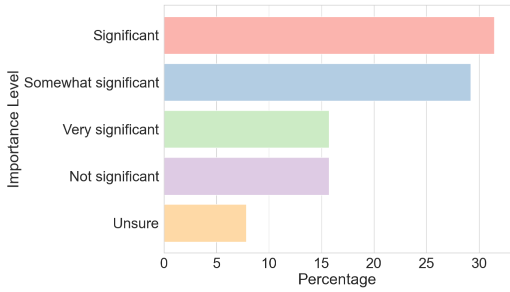


Figure 4.20: *Participants' Satisfaction Ratings Regarding AI Tools*



Finally, participants were asked about the perceived importance of prior knowledge in programming or databases for effectively using AI tools. According to Figure 4.21, 76.4% of respondents (combining "Very significant," "Significant," and "Somewhat significant") acknowledged the importance of such knowledge to varying degrees. In contrast, 15.7% considered it "Not significant," and 7.9% were "Unsure." This distribution suggests that while a majority recognize the value of foundational technical skills in leveraging AI tools effectively, a notable portion believes that prior knowledge may not be essential.

Figure 4.21: *Participants' Perceptions of the Role of Prior Knowledge in AI Tool Usage*



Overall, this section provides a comprehensive overview of participants' interaction patterns with AI tools, their trust levels, satisfaction, and perceptions regarding the relevance of technical background. The findings indicate a generally positive stance towards AI tools, coupled with an awareness of the value of prior technical knowledge.

In the continuation of this section, the analysis will focus on the four research questions that constitute the core framework of this study by combining insights from multiple survey items.

4.2 Research Questions Analysis

This section presents the empirical findings in relation to the defined research questions. Each question is addressed through descriptive and comparative analysis based on the survey data, aiming to reveal patterns and insights that inform the role of programming and OOP skills in engineering-related professional contexts.

4.2.1 Research Question 1

The first research question explores the role and perceived importance of programming knowledge particularly OOP among engineering professionals. It examines self-assessed programming proficiency, how critical such skills are for one's role, familiarity with OOP concepts, and whether lack of OOP knowledge has impacted job effectiveness. Sector- and department-level breakdowns provide insight into where programming skills are most prevalent and valued.

- **Q1:** How would you rate your overall programming knowledge?

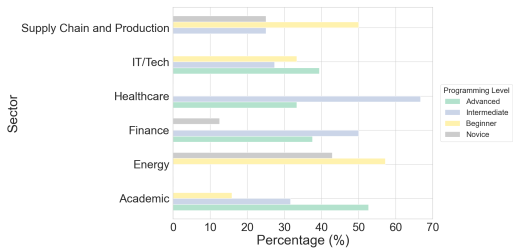
  First, self-reported programming proficiency levels (Q1) were analyzed across the six most represented sectors and departments in the dataset. The results reveal clear variation across contexts.

  As shown in Figure 4.22, the academic sector stands out, with a majority of respondents identifying as either advanced (52.6%) or intermediate (31.6%). This aligns with the research-intensive nature of academic roles, which often demand technical and programming expertise.

  The finance and IT/tech sectors also report high proficiency: in finance, 37.5% of respondents considered themselves advanced and 50% intermediate; in IT/tech, 39.4% were advanced and 27.3% intermediate. These roles frequently involve data processing, algorithms, or systems integration, which likely explains the emphasis on programming.

  By contrast, the energy and supply chain/production sectors show notably lower levels of programming knowledge. In energy, all respondents rated themselves as beginners (57.1%) or novices (42.9%). Similarly, in supply chain/production, half of the respondents were beginners, and 25% novices. These results suggest that programming is less embedded in day-to-day operations in these fields.

Figure 4.22: *Programming Knowledge Level by Top 6 Sectors*

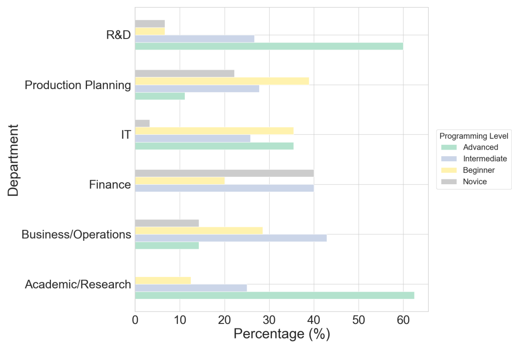


Looking at departments (see Figure 4.23), academic/research and R&D departments report the highest programming skills, with over 60% of respondents identifying as advanced in both. These departments typically involve computational modeling, simulations, or development of custom tools.

In contrast, departments such as finance, business/operations, and production operations planning had higher shares of beginners and novices. For instance, no respondent in the finance department rated themselves as advanced, and 60% were in the beginner or novice categories. Similarly, in business/operations, 28.6% were beginners and 14.3% novices.

Interestingly, programming skills are also uneven within IT departments: while 35.5% rated themselves as advanced, an equal percentage reported being beginners. This likely reflects the diversity of roles within IT—some focused on development, others on infrastructure or support.

Figure 4.23: *Programming Knowledge Level by Top 6 Departments*



Overall, the findings underscore that programming proficiency varies widely across engineering roles, shaped by the technical demands of each sector and department. These differences point to the need for targeted training strategies that align with job-specific skill requirements.

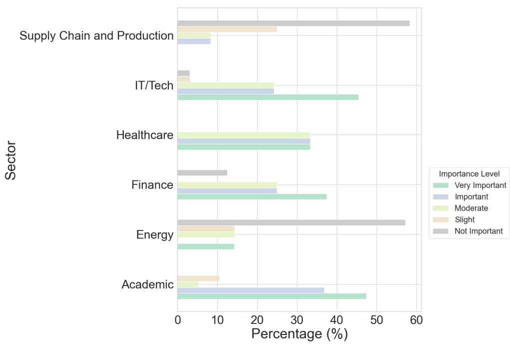
- **Q5:** How critical is programming knowledge for your role?

Beyond self-assessed proficiency, Research Question 1 also examines the perceived importance of programming knowledge (Q5), shedding light on how essential these skills are considered across different sectors and departments. This perspective helps clarify whether programming is viewed as a core, complementary, or negligible skill depending on the professional context.

As shown in Figure 4.24, the academic (47.4%) and IT/Tech (45.5%) sectors had the highest proportions of respondents rating programming knowledge as "very important." These results align with expectations, as both sectors frequently involve tasks such as data analysis, modeling, and system development that inherently rely on programming. The finance sector followed with 37.5% rating it as "very important," reflecting the increasing role of automation and quantitative modeling in financial services.

In contrast, the energy and supply chain/production sectors placed significantly lower emphasis on programming skills. In the supply chain/production domain, none of the respondents considered programming to be "very important," and 58.3% found it "not important" at all. Similarly, 57.1% of energy sector respondents also viewed programming as "not important." This trend may reflect the more operational and hardware-focused nature of tasks in these fields, where programming is less embedded in daily workflows.

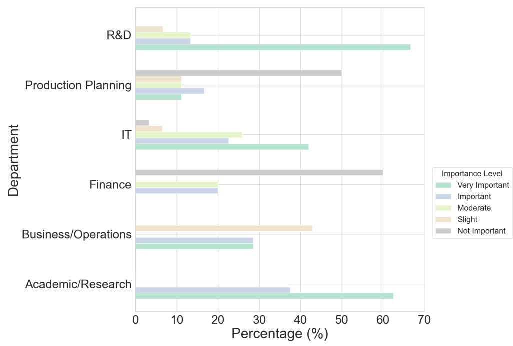
Figure 4.24: *Perceived Importance of Programming by Top 6 Sectors*



A similar pattern is observed when the data is viewed by department (Figure 4.25). R&D and academic/research departments stood out with over 60% of participants describing programming as "very important." These departments often require custom tool development, simulation, and advanced data analysis—activities for which programming is integral. On the other hand, finance and production operations planning departments showed a marked undervaluation of programming: 60% of finance respondents and 50% of production planning respondents stated that programming is "not important" for their roles.

Interestingly, the IT department revealed a wide distribution: 41.9% rated programming as "very important," but 6.5% rated it as "slight" and 3.2% as "not important." This once again highlights the diversity of roles within IT departments, where not all functions (e.g., network administration, support) demand the same level of programming engagement as software development roles do.

Figure 4.25: *Perceived Importance of Programming by Top 6 Departments*



Therefore, the findings suggest that programming is perceived as critical primarily in research-intensive and tech-driven environments. In contrast, its perceived relevance drops sharply in operational and traditional engineering sectors, underscoring the importance of role-specific training and upskilling strategies.

- **Q7:** How would you rate your knowledge of OOP concepts?

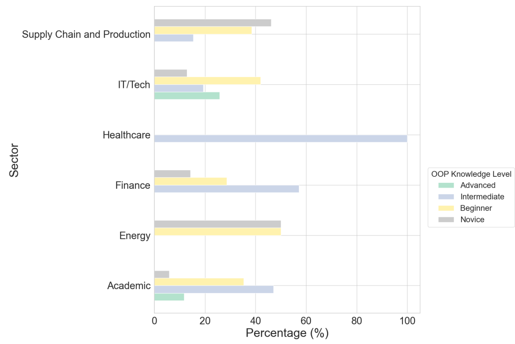
This question assesses the respondents' self-reported proficiency in OOP, which is a fundamental aspect of programming education and practice. The four proficiency levels—Advanced, Intermediate, Beginner, and Novice—provide insight into the depth of participants' programming knowledge across sectors and departments.

An analysis by sector reveals marked disparities in self-assessed proficiency levels (Figure 4.26). As expected, respondents from the IT/Tech sector exhibited the highest proportion of Advanced-level competence (25.8%), reflecting the centrality of programming, and specifically OOP, in the typical responsibilities associated with this domain. In contrast, no respondents from the Healthcare, Finance, Energy, or Supply Chain/Production sectors reported an Advanced level of proficiency. This observation points to a potential gap in either the perceived relevance of OOP or in training opportunities within these sectors.

Nevertheless, the data also suggest that certain non-technical sectors maintain a moderate familiarity with OOP concepts. For instance, all respondents from the Healthcare sector reported Intermediate-level proficiency (100.0%), indicating a uniform but moderate degree of exposure. Similarly, a majority of respondents from the Finance sector (57.1%) also assessed themselves as Intermediate, with the remainder split between Beginner and Novice levels. The Supply Chain and Production sector, however, demonstrated the lowest overall proficiency, with a significant portion identifying as Novice (46.2%) or Beginner (38.5%).

The Academic sector displayed a more heterogeneous distribution, with 11.8% reporting Advanced proficiency, 47.1% Intermediate, 35.3% Beginner, and 5.9% Novice. This variation may be attributed to the diversity of academic disciplines and the degree to which programming is integrated into research and instruction.

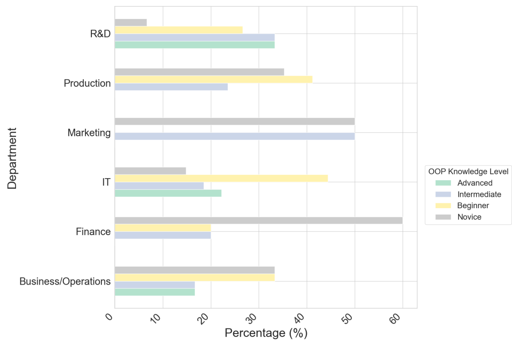
Figure 4.26: *OOP Knowledge Level by Top 6 Sectors*



A departmental breakdown (Figure 4.27) further emphasizes the variation in proficiency levels. Respondents from R&D and IT departments reported the highest levels of expertise, with 33.3% and 22.2% indicating Advanced proficiency, respectively. The Business department followed with a smaller share (16.7%). In contrast, Finance, Marketing, and Production departments did not report any Advanced-level respondents, and instead, showed a strong presence of Beginner and Novice levels. For example, 60.0% of Finance respondents and 50.0% of Marketing respondents assessed themselves as Novice.

It is also noteworthy that the IT department, while expected to demonstrate higher proficiency overall, exhibited a wide distribution. Alongside the 22.2% reporting Advanced competence, 44.4% identified as Beginner and 14.8% as Novice. This suggests that IT functions encompass a broad spectrum of roles, not all of which require intensive programming engagement.

Figure 4.27: *OOP Knowledge Level by Top 6 Departments*



In sum, the data underscore substantial differences in OOP proficiency across both sectors and departments. While higher levels of expertise are concentrated in technically oriented domains, many non-technical roles demonstrate limited engagement with OOP. These findings highlight the need for role-specific capacity-building initiatives, particularly in departments that are increasingly expected to collaborate with technical teams or to engage with digital tools requiring programming literacy.

- **Q10:** Have you encountered situations where lack of proficiency in OOP limited your effectiveness?

  This question aimed to investigate whether insufficient knowledge of OOP principles posed a barrier to professional performance, particularly across different sectors and departmental functions.

  The aggregated responses, visualized in Figure 4.28 and Figure 4.29, demonstrate a notable variation depending on the professional setting. Figure 4.28, which presents sector-based results, professionals in the Healthcare and IT/Tech sectors reported the highest perceived negative impact, with 33.3% and 32.3% respectively indicating that a lack of OOP proficiency had hindered their effectiveness. Notably, while the Healthcare sector is not typically associated with intensive OOP use, a closer examination reveals that respondents from this group were employed in analytically oriented roles such as Senior Business Process Engineer, Research Data Analyst, and Senior Healthcare Systems Engineer. Furthermore, they were positioned in departments such as Production Operations Planning, R&D, and the Office of Performance Improvement, where computational and system-oriented thinking is more likely to be required. Combined with the relatively small number of respondents from this sector, these contextual factors help explain the elevated perceived impact within Healthcare. In contrast, fields such as Finance and Energy exhibited minimal concern, with the majority of respondents selecting either "No" or "Does Not Apply," implying that OOP concepts are less integral to their day-to-day tasks.

Figure 4.28: *Perceived OOP Knowledge Effectiveness by Sector*

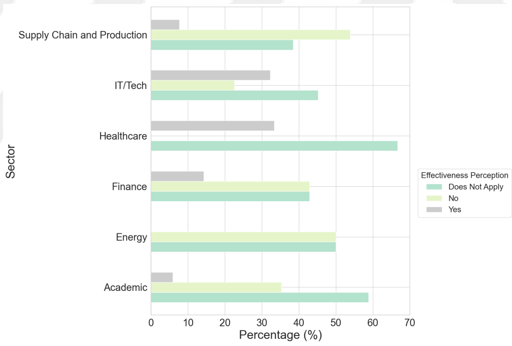
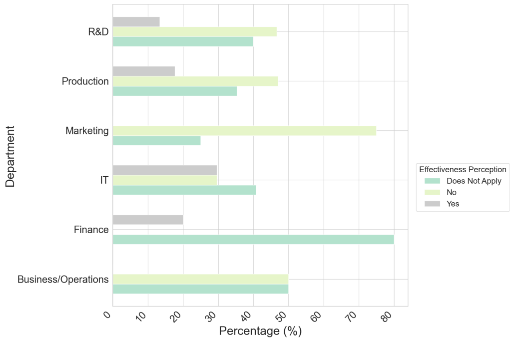


Figure 4.29 displays departmental-level findings and supports similar conclusions. Within the IT department, 29.6% of respondents affirmed that insufficient OOP knowledge had negatively influenced their performance. Meanwhile, departments such as Finance and Marketing showed very limited dependency on OOP skills; for example, 80% of respondents in Finance selected "Does Not Apply," suggesting that such knowledge is largely irrelevant in that context.

Figure 4.29: *Perceived OOP Knowledge Effectiveness by Department*



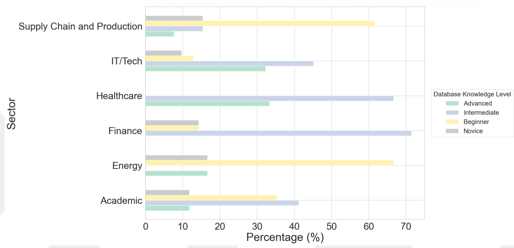
These observations collectively reinforce the domain-specific importance of OOP literacy. While critical in technical roles that involve software design or system architecture (e.g., IT and R&D), it is considerably less relevant in business-facing functions such as marketing or finance. Therefore, tailored training initiatives may be more impactful when aligned with specific sectoral or departmental needs.

Since this research stems from the broader context of Software Engineering Fundamentals, it does not solely focus on programming knowledge but also considers foundational topics such as database systems. Therefore, database-related knowledge was also examined through Questions 12 and 15 to provide a more comprehensive understanding of technical proficiency across sectors and departments.

- **Q12:** How would you rate your knowledge of Database concepts?

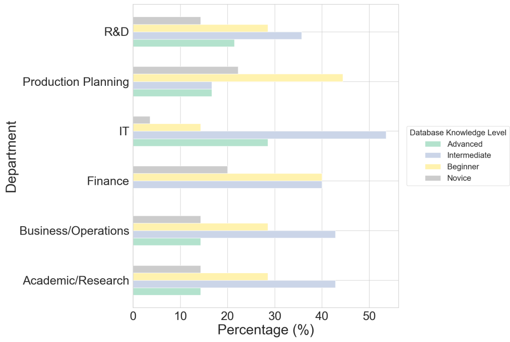
  The analysis of database knowledge distribution across different sectors reveals notable variations in proficiency levels. As illustrated in Figure 4.30, the Healthcare and IT/Tech sectors exhibit the highest proportions of respondents who consider themselves advanced in database concepts, with 33.3% and 32.3% respectively. Also, a significant portion of participants from these sectors have indicated that they are at an intermediate level. This likely reflects the technical nature of roles within these sectors, where database management and optimization are integral. In contrast, the Finance sector lacks respondents identifying as advanced but shows a predominant intermediate-level proficiency at 71.4%. The Academic sector demonstrates a more balanced distribution across proficiency levels, with a significant share of respondents rating themselves as intermediate or beginner. Conversely, sectors such as Energy and Supply Chain and Production have larger shares of beginners and novices, indicating a relatively lower emphasis on database expertise in these fields.

Figure 4.30: *Perceived Database Knowledge by Sector*



When examining database knowledge by department Figure 4.31, the IT department stands out with the largest share of advanced-level respondents at 28.6%, followed by R&D and Production Planning departments. Intermediate proficiency is widespread across IT, Academic/Research, and Business/Operations departments, highlighting a generally solid foundation in database concepts within these groups. However, the Finance department again shows no advanced-level expertise, instead having a considerable portion of beginners and novices combined, which suggests potential areas for targeted training interventions. Similarly, the Production Planning department has a notable proportion of respondents with limited database knowledge.

Figure 4.31: *Perceived Database Knowledge by Department*



These findings underscore the heterogeneous nature of database proficiency across sectors and departments. While technical departments and sectors demonstrate higher competency levels, others reflect opportunities for skill development. Such disparities highlight the importance of designing tailored training programs that address the specific needs of different professional contexts, ensuring that database knowledge aligns with the operational demands of each sector and department.

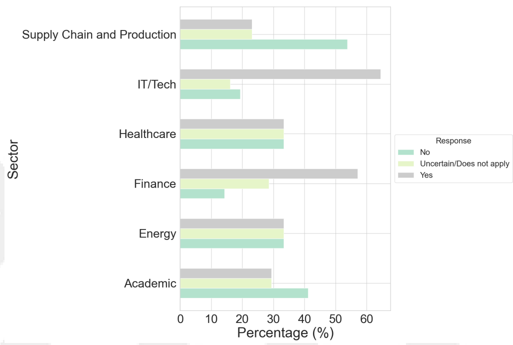
- **Q15:** Have you encountered situations where the lack of proficiency in database concepts limited your effectiveness?

  As illustrated in Figure 4.32, responses vary notably across sectors. The IT/Tech sector stands out, with 64.5% of participants reporting that insufficient database knowledge had indeed limited their performance. This is perhaps unsurprising, given the centrality of data systems in software development, system administration, and related roles. A similar trend is observed in the finance sector (57.1%), where database fluency is often required for tasks such as query-based reporting, data auditing, or integration with financial systems.

  In contrast, sectors such as supply chain/production and academic institutions report the lowest percentages of adverse impact, with 53.8% and 41.2% respectively indicating no such experience. This suggests that in these environments, database work may either be less central to day-to-day responsibilities or sufficiently supported by specialized staff.

Across all sectors, a non-trivial portion of respondents selected Uncertain/Does not apply, particularly in energy (33.3%) and healthcare (33.3%), which may indicate varying interpretations of “effectiveness” or reflect roles with marginal exposure to database tasks.

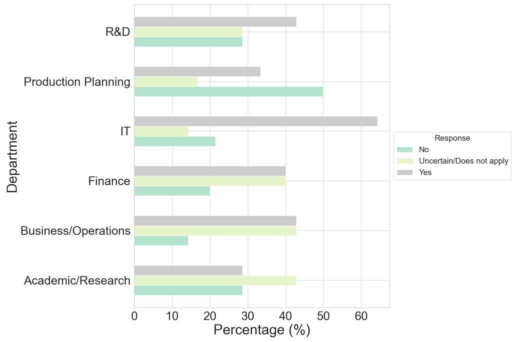
Figure 4.32: *Impact of Lack of Database Knowledge by Top 6 Sectors*



A parallel pattern emerges when analyzing department-level data (Figure 4.33). Within IT departments, nearly two-thirds of respondents (64.3%) confirmed that insufficient database knowledge had hindered their work—reinforcing the critical role of databases in IT functions. Similarly, high affirmative responses were reported in R&D (42.9%) and business/operations departments (42.9%), where data systems often intersect with analysis, reporting, or process design tasks.

Interestingly, academic/research departments were less affected (28.6% reported ‘Yes’), but showed a strong representation in the Uncertain/Does not apply category (42.9%), perhaps due to varied backgrounds in the respondent pool. Production planning had the highest share of respondents reporting No (50%), further suggesting that in certain operational contexts, database fluency may not be mission-critical.

Figure 4.33: *Impact of Lack of Database Knowledge by Top 6 Departments*



Together, these findings highlight that the need for database proficiency is unevenly distributed across roles. While IT, finance, and analytical departments tend to experience stronger dependencies, other sectors may rely more on prebuilt tools or delegated support, reducing the direct impact of database knowledge gaps.

The overall results reveal clear variations in programming and database knowledge across sectors and departments, reflecting differences in role requirements and exposure to technical skills. The IT/Tech sector consistently reports the highest levels of advanced and intermediate proficiency in both OOP and database concepts, underscoring the technical nature of its roles. The Academic sector shows relatively high familiarity with programming concepts but a more balanced and moderate proficiency distribution in database knowledge, with many respondents rating themselves as intermediate or beginner rather than advanced. Conversely, sectors such as Healthcare, Finance, Energy, and Supply Chain/Production generally exhibit lower proficiency levels, with a larger share of beginners and novices, particularly in programming skills. The perceived impact of insufficient knowledge also differs by sector: IT/Tech and Healthcare professionals more often indicate that gaps in programming or database proficiency have limited their effectiveness, while Finance, Marketing, and Energy respondents frequently view these skills as less central to their daily responsibilities.

Departmental analysis aligns with these trends, highlighting R&D and IT departments as areas of higher competency, with Finance, Marketing, and Production departments displaying lower proficiency and less dependency on these technical skills. Even within IT departments, a wide range of skill levels exists, reflecting diverse job roles. The results indicate that deficiencies in programming and database knowledge, varying by sector and department, may negatively impact professionals' performance, pointing to the critical need for targeted training programs tailored to these specific contexts.

4.2.2 Research Question 2

To complement the skill-based findings presented earlier, Research Question 2 focuses on engineers' reflections regarding the adequacy of their university education in preparing them for real-world technical challenges. This section investigates how graduates evaluate their academic foundations across different professional environments. Drawing on responses to Questions 6, 9, and 14, the analysis captures variations in perceived preparedness based on country of education, as well as the technological sophistication of their current department and sector. This approach enables a contextualized understanding of whether formal education aligns with the evolving technical demands of contemporary engineering roles.

- **Q6:** How adequate was the programming knowledge you gained in university for your professional life?

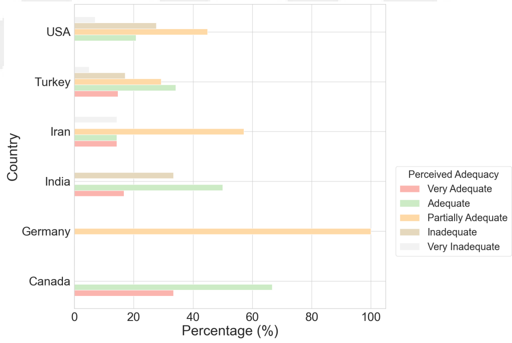
  – Country-Based Variation:

    As illustrated in Figure 4.34, perceptions of programming education adequacy vary significantly across countries. To ensure meaningful comparisons, the figure includes the six countries with the highest number of responses. Respondents educated in Canada reported the highest satisfaction, with 100%

rating their education as Adequate or Very Adequate. In contrast, all German participants found their education only Partially Adequate. Graduates from Turkey—the largest group in the sample—showed a mixed distribution, with responses spread across Very Adequate (14.6%), Adequate (34.1%), and Partially Adequate (29.3%). Similarly, participants from Iran, India, and the USA predominantly rated their education as moderate or below, with particularly critical views observed among US respondents, 34.5% of whom found their education Inadequate or Very Inadequate.

These cross-national patterns suggest that institutional and curricular differences may influence how well university programming education aligns with professional expectations.

Figure 4.34: *Perceived Adequacy of University Programming Education by Country*



– Sector and Department-Based Variation: This cross-country variation is further complemented by patterns observed across professional sectors and departments, as shown in Figure 4.35 and Figure 4.36. To ensure representativeness, the figure focuses on the six most frequently reported sectors and departments.

Participants working in academia and healthcare generally expressed higher satisfaction with their university programming education, with over 60% of respondents in both groups rating it as Adequate or Very Adequate. In contrast, those in energy and supply chain and production sectors showed lower levels of satisfaction, with higher shares selecting Inadequate or Very Inadequate.

From department perspective, R&D employees reported the highest perceived adequacy, with 56.2% rating their background as Adequate or Very Adequate. Conversely, respondents in production and IT departments showed more critical views, with over 25% rating their education as Inadequate or worse. These patterns indicate that perceptions of programming education adequacy may vary depending on the technical expectations and contextual demands of different sectors and departments. Conversely, the nature and quality of undergraduate programming instruction might also influence individuals' later specialization choices, steering them toward certain professional domains over others.

Figure 4.35: *Perceived Adequacy of University Programming Education by Sector*

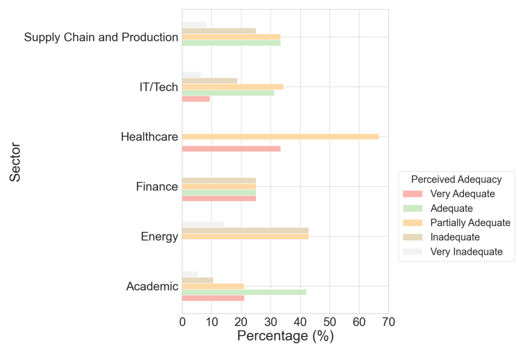
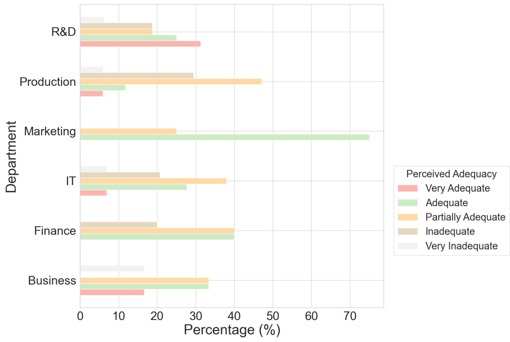


Figure 4.36: *Perceived Adequacy of University Programming Education by Department*



To further explore the origins of these skill gaps, the next question investigates whether participants perceived their university education as sufficient in preparing them to apply OOP concepts to real-world industry challenges.

- **Q9:** Was your university education sufficient for applying OOP in real-world industry problems?
  - Country-Based Variation: As illustrated in Figure 4.37, perceptions of OOP preparation vary notably across the top six countries in the sample. Canada stands out with the most positive view—two-thirds of its respondents found their education sufficient, and the rest selected ”Partially”. In contrast, Germany showed a more cautious stance, with 100% of participants choosing ”Partially”.

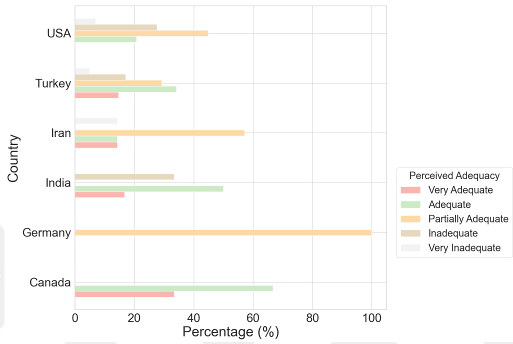
    India and Iran revealed more mixed patterns. While most Indian respondents found their education only partially sufficient, a small portion (16.7%) selected ”Yes”, and another 16.7% considered it insufficient. Iranian participants were evenly split between ”Partially” and ”No”, with no one selecting ”Yes”.

    Among Turkish graduates—the largest national group—only 5.3% felt fully prepared for OOP, while nearly two-thirds chose ”Partially” and around 32%

selected "No". The USA showed the most critical view, with 62.1% of respondents stating that their education did not prepare them adequately for OOP in real-world applications.

These differences highlight how national variations in curricula and teaching focus may impact graduates' confidence in applying OOP professionally.

Figure 4.37: *Country-Based OOP Education Sufficiency*



– Sector and Department-Based Variation: As shown in Figures 4.38 and 4.39, perceptions of how well university education prepared participants for applying OOP differ noticeably across sectors and departments. In sectoral terms, healthcare respondents reported the most positive experiences, with one-third stating their education was sufficient. In contrast, views were more critical in the energy and finance sectors, where a significant share of participants considered their training inadequate.

  At the departmental level, IT professionals (who typically engage more directly with OOP) reported the lowest satisfaction, only 3.7% found their education sufficient, while over 40% rated it as inadequate. Departments like business and production showed a more balanced distribution, while responses from marketing and R&D participants tended to cluster around partial sufficiency.

Figure 4.38: *Sector-Based OOP Education Sufficiency*

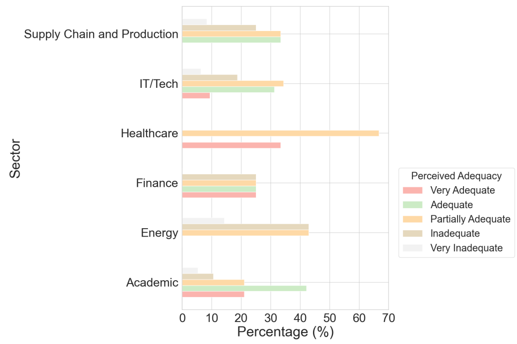
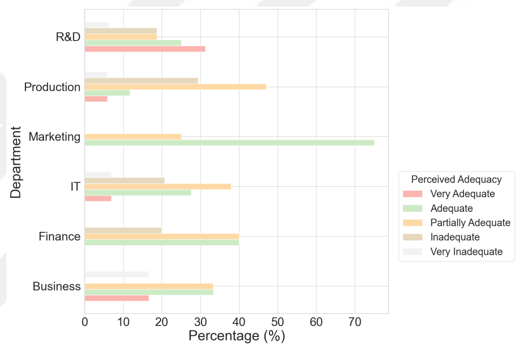


Figure 4.39: *Department-Based OOP Education Sufficiency*



To provide a broader view of perceived educational sufficiency, the following question shifts the focus from programming to database concepts and explores how university instruction in this domain is evaluated across similar professional dimensions.

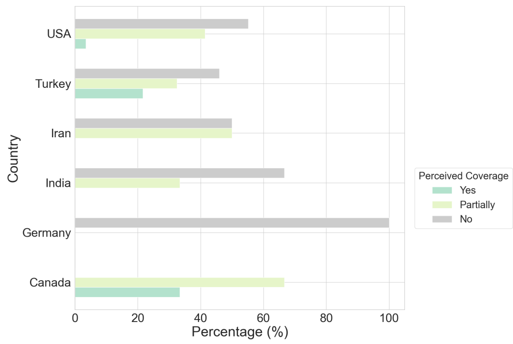
- **Q14:** Do you feel that your university education sufficiently covered Database concepts and their applications?

  – Country-Based Variation: As shown in Figure 4.40, the adequacy of database education during university varies noticeably by country. Among the six countries with the highest response counts, Canada stands out with relatively favorable evaluations: 33.3% of Canadian graduates felt their education sufficiently covered database concepts, and 66.7% selected Partially. In contrast, participants from Germany expressed unanimous dissatisfaction, with 100% responding No. Similarly, the vast majority of respondents from India, Iran, and the USA indicated insufficient coverage, with more than half of U.S. participants also selecting No.

Turkey, which again comprises the largest national group, showed a more balanced distribution: 21.6% reported adequate coverage, while nearly half (45.9%) found their database education lacking. However, it is important to note that a portion of Turkish respondents who answered Yes hold undergraduate degrees in CS or Mathematics Engineering rather than ISE. Specifically, among the 8 Turkish respondents who evaluated their database education as sufficient, only 5 were ISE graduates, while the rest came from more technically focused disciplines. Conversely, among the 29 Turkish participants who selected No or Partially, 23 held ISE degrees. This disparity highlights a potential gap in database-related instruction within ISE curricula in Turkey, suggesting that graduates from this field may not be adequately equipped in this domain compared to their peers from computer-oriented programs.

These results reinforce the idea that country-level curricular standards and the degree of applied database instruction play a critical role in shaping graduates' perceptions of preparedness.

Figure 4.40: *Country-Based Database Education Sufficiency*



– Sector and Department-Based Variation: Figures 4.41 and 4.42 present how participants from various sectors and departments assess the extent to which their university education covered database concepts and applications. Responses reveal a notable degree of variation across professional contexts. For instance, although the IT/Tech sector reported the highest proportion of affirmative responses (19.4%), it also had a considerable share of negative evaluations (51.6%). This contrast may reflect two related patterns: those with stronger database education might be more likely to enter technical roles, while people in such roles may judge their academic training more critically because of the higher demands they face in practice.

  A similar pattern is observed at the departmental level. IT professionals were more likely than others to state that their education was sufficient, yet nearly half disagreed. Participants from Finance and Production departments also tended to report limited coverage, while R&D respondents largely selected the "Partially" option. These findings suggest that individuals in more technical roles may both benefit from, and critically evaluate, their academic exposure to database-related content.

  These patterns do not indicate absolute sufficiency or deficiency, but rather suggest a nuanced relationship between academic preparation and the data-related demands of different work settings.

Figure 4.41: *Sector-Based Database Education Sufficiency*

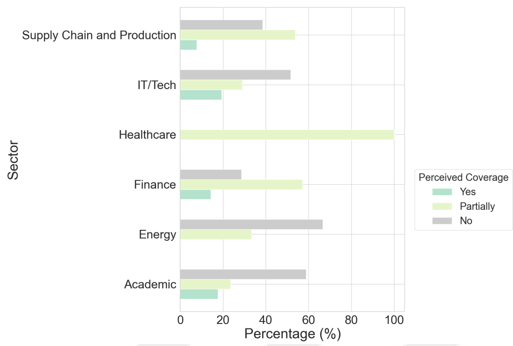
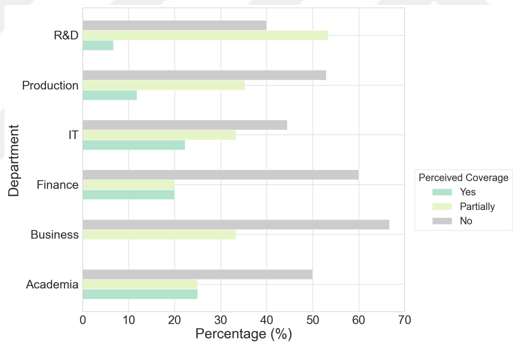


Figure 4.42: *Department-Based Database Education Sufficiency*



Overall, the findings from Research Question 2 show that while university education offers a basic foundation in programming and database concepts, perceptions of its sufficiency vary considerably depending on both where graduates received their education and the context in which they currently work. Participants educated in countries like Canada tend to view their academic preparation more positively, whereas those from countries such as Germany and the USA more often report shortcomings. Similar differences appear across sectors and departments: individuals in technical roles such as IT or R&D are often more critical, possibly because their roles demand greater practical application of these skills. These patterns suggest that aligning academic curricula more closely with the technical realities of modern engineering roles could better support students in their transition to the workplace.

Building on insights from OOP proficiency, this section examines database skills through Questions 15 and 16. These explore whether lack of database knowledge has limited professional effectiveness and which career aspects benefit from strong database proficiency, offering a complementary view on technical competencies in engineering careers.

4.2.3 Research Question 3

Building on the previous analyses of technical skills, Research Question 3 examines how proficiency in OOP and database concepts influences engineers' professional development, project success, and problem-solving abilities. This section draws on survey responses related to the impact of these competencies on career outcomes, providing a broader perspective on how such skills contribute to professional effectiveness across different engineering contexts.

- **Q10:** Have you encountered situations where the lack of proficiency in OOP concepts limited your effectiveness?

  As previously detailed in Overview of Survey Findings subsection, the impact of insufficient OOP skills varies by sector and role. Overall, Figure 4.9 shows that 16.7% of respondents experienced negative effects on their professional effectiveness due to limited OOP proficiency, while 37.8% did not perceive such limitations. Notably, a substantial 45.6% selected "Uncertain/Does not apply," underscoring that the relevance of OOP skills is not uniform across all engineering positions.

- **Q11:** Which aspects of your career have been positively influenced by proficiency in OOP concepts?

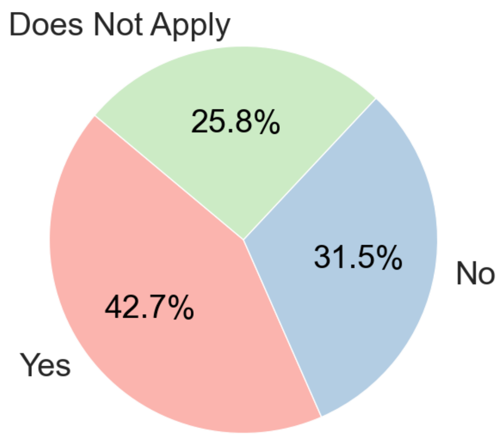
  Building on these findings, Question 11 examines the perceived benefits of OOP proficiency on career outcomes. As illustrated in Figure 4.11, the most commonly reported advantage is enhanced problem-solving ability, followed by project success and improved team collaboration. Fewer respondents associated OOP skills with promotion opportunities or reported no noticeable impact. The high proportion selecting "Uncertain/Does not apply" further suggests that the value of OOP skills depends heavily on individual job functions and contexts.

  Taken together, these results emphasize that while OOP proficiency is not universally essential, it plays a significant role in supporting core technical competencies such as problem-solving and project success within specific professional settings.

- **Q15:** Have you encountered situations where the lack of proficiency in database concepts limited your effectiveness?

  Figure 4.43 illustrates how limitations in database knowledge can similarly hinder professional effectiveness. According to the responses, 42.7% of participants have encountered situations where a lack of database proficiency negatively impacted their work. While 31.5% reported no such experience, 25.8% selected "Does Not Apply," suggesting that database-related tasks may not be central to all engineering roles. These findings parallel the trends observed for OOP, emphasizing that technical fluency—whether in programming or databases—can influence an engineer's ability to perform effectively depending on the nature of their responsibilities.

Figure 4.43: *Impact of Database Proficiency Gaps on Professional Effectiveness*

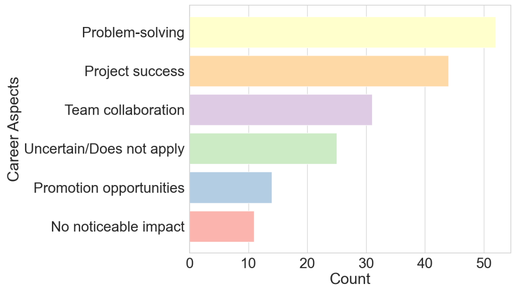


To complement this observation, Question 16 explores the perceived career benefits associated with stronger database proficiency.

- **Q16:** Which aspects of your career have been positively influenced by proficiency in Database concepts?

As shown in Figure 4.44, database proficiency is most frequently associated with enhanced problem-solving and project success, followed by notable mentions of team collaboration. Fewer respondents linked these skills to promotion opportunities or reported no noticeable impact. A considerable number selected "Uncertain/Does not apply," reflecting variation in how database skills relate to different roles. Overall, the responses suggest that database fluency primarily supports core technical activities rather than broader career outcomes.

Figure 4.44: *Positive Impacts of Database Proficiency on Professional Effectiveness*



Overall, the findings from Research Question 3 suggest that proficiency in OOP and database concepts may contribute positively to engineers' professional effectiveness, particularly in areas such as problem-solving and project success. While a substantial number of participants reported limited impact or uncertainty—especially regarding OOP those with stronger technical skills more frequently associated them with beneficial career outcomes. These observations indicate a potential link between technical fluency and improved performance, though the variation in responses also highlights that the relevance of such skills can differ depending on role-specific demands and professional contexts.

4.2.4 Research Question 4

To better understand the evolving role of AI in engineering practice, Research Question 4 examines how engineers use AI-based tools in daily work and decision-making. It explores the frequency and purpose of usage, trust in outputs, and whether prior programming knowledge affects effective use. By analyzing responses across departments, this section aims to show how technical background and professional context shape AI tool integration.

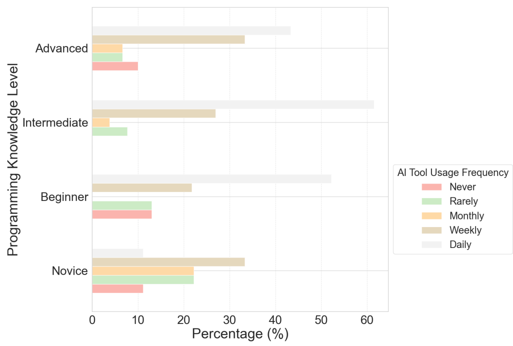
- **Q17:** How often do you use ChatGPT or similar AI-based tools?

This question assesses how often engineers use AI tools. The analysis by programming proficiency and department reveals patterns in usage related to skill levels and work contexts.

As Figure 4.45 shows, daily AI tool usage is highest among intermediate (61.5%) and beginner (52.2%) programmers. Novices also use these tools regularly, with about one-third engaging weekly.

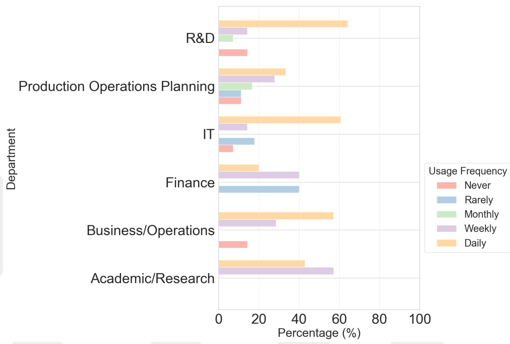
Meanwhile, advanced programmers demonstrate a more distributed pattern, with 43.3% using these tools daily and 33.3% weekly. This may suggest that while advanced users are actively engaging with AI tools, they may also be more selective in their use cases, or rely on a broader technical toolkit. Overall, the data points to a strong relationship between baseline programming knowledge and the frequency of AI tool adoption, with even basic proficiency appearing sufficient to support regular use.

Figure 4.45: *Frequency of ChatGPT Use Across Different Programming Backgrounds*



These departmental patterns, illustrated in Figure 4.46, suggest that the frequency of AI tool usage may be shaped not only by individual proficiency levels, but also by the nature of tasks commonly performed within different organizational roles. Daily engagement appears more likely in departments such as R&D and IT, where problem-solving and experimentation are central to daily workflows. In contrast, departments like Finance and Production exhibit more moderate usage patterns, which could reflect either lower demand for generative assistance or a greater reliance on domain-specific tools. Although not conclusive, these variations indicate that the type of tasks within different departments might affect how often engineers use AI tools in their work.

Figure 4.46: *Frequency of ChatGPT Use Across Different Departments*

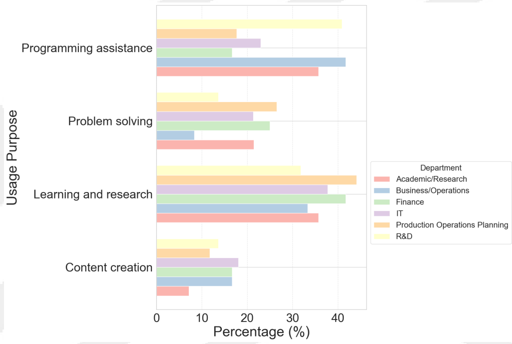


- **Q18:** For what purposes do you use ChatGPT or similar AI tools?

  This question examines the specific ways engineers employ AI-based tools in their work. Given that usage is expected to vary by department, the analysis centers on departmental differences to highlight how AI supports diverse professional areas.

  Across all departments, learning and research emerge as the predominant purposes for using AI-based tools. Departments like Academic/Research and Production Operations Planning particularly emphasize these activities, reflecting their focus on knowledge acquisition and continuous development. Meanwhile, Business/Operations and R&D departments show a relatively higher reliance on AI for programming assistance, problem solving, and content creation. This suggests that professionals in these areas may engage more frequently with technical tasks or programming-related challenges. Marketing and Finance demonstrate a more balanced distribution of AI usage purposes, indicating diverse application scenarios. Overall, these patterns highlight how the specific demands and expertise of each department influence the integration of AI tools into engineers' daily work and decision-making processes.

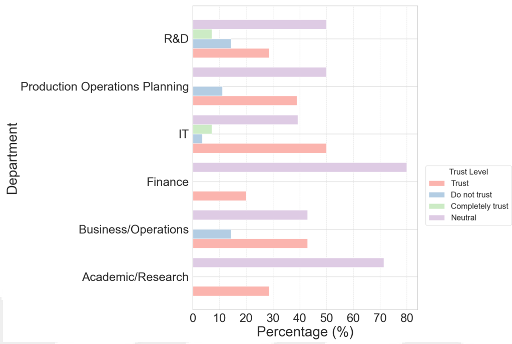
Figure 4.47: *Departmental Variation in the Purposes of Using AI-Based Tools*



- **Q19:** How much do you trust the outputs provided by ChatGPT or similar AI-based tools?

  To evaluate the perceived reliability of AI-generated outputs within engineering domains, respondents were asked to indicate their level of trust in tools. As illustrated in Figure 4.48, responses reveal a predominantly cautious stance across departments. Neutrality emerges as the most common attitude, particularly within Finance (80%) and Academic/Research (71.4%), suggesting a measured approach in accepting AI-generated content. Although the IT department displays the highest combined proportion of trust and complete trust (57.1%), this still reflects a moderate level of confidence rather than unequivocal endorsement. Departments such as Business/Operations and R&D exhibit a more varied distribution across trust levels, indicating diverse perspectives shaped by differing use cases and expectations. Overall, the findings imply that while engineers are increasingly engaging with AI tools, their confidence in these systems' reliability remains reserved.

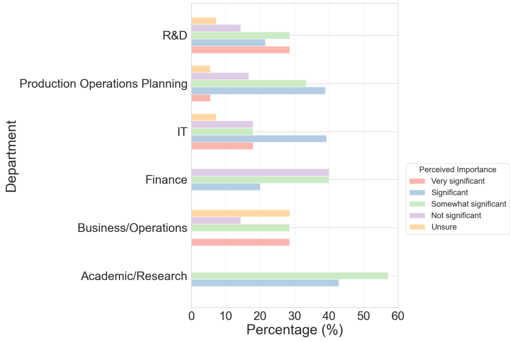
Figure 4.48: *Departmental Variation in Trust Levels Toward AI-Based Tools*



– **Q21:** How significant do you believe prior knowledge in programming or databases is for effectively utilizing ChatGPT or similar AI tools?

Understanding how engineers perceive the importance of prior knowledge and experience related to the effective use of AI-based tools is crucial for assessing their integration into daily workflows. Figure 4.49 presents these perceptions across departments, revealing that the majority of respondents view such knowledge as either "Significant" or "Somewhat significant." Departments like IT and Production Operations Planning exhibit the highest recognition of this importance, with over 70% of participants emphasizing the value of prior experience, reflecting the technical demands of their roles. Conversely, departments such as Finance and Business/Operations show more varied opinions, including higher proportions of "Not significant" and "Unsure" responses. This variation highlights how the perceived relevance of prior knowledge depends strongly on the specific functions and requirements of each department.

Figure 4.49: *Departmental Perception of the Importance of Prior Technical Knowledge for Effective AI Tool Usage*



The analysis of Research Question 4 highlights a complex and department-specific pattern in how engineers interact with AI-based tools like ChatGPT. The frequency of usage is closely linked to both programming skills and departmental roles, with technical areas such as IT and R&D showing notably higher daily engagement, likely due to the problem-solving and experimental nature of their work. The purposes for using AI tools—ranging from learning and research to programming support and content creation—also differ across departments, mirroring their unique professional demands. Trust in AI-generated outputs tends to be moderate, with many respondents in Finance and Academic/Research adopting a neutral stance, while IT professionals show slightly more confidence, though not overwhelmingly so. Importantly, the perceived significance of prior technical knowledge for effective AI tool use varies by department, with technical teams emphasizing its necessity and other units displaying more diverse views. Together, these insights suggest that successful AI integration in engineering depends on a balance of individual expertise, departmental context, and task requirements, underscoring the importance of tailored training and adoption strategies that reflect these factors.

4.3 Analytical and Statistical Framework

To complement the descriptive summaries, key results of the research questions are examined from an analytical and statistical perspective. For outcomes that are continuous or can be treated as approximately interval-scaled, group differences were evaluated primarily with independent-samples Welch *t*-tests, which are appropriate under unequal variances and unbalanced group sizes. For each contrast, the test statistic, associated degrees of freedom, two-tailed *p*-value (α=0.05), an effect size suitable to the test (e.g., Cohen's d for mean differences), and the 95% confidence interval (CI) are reported.

4.3.1 Research Question 1: Programming Knowledge

- **Key Result:** Programming competence is important in academic roles alongside strictly technical roles.
- **Scale & test:** Programming level was measured on a 1–4 scale (Novice=1, Beginner=2, Intermediate=3, Advanced=4) and compared across departments using independent-samples Welch *t*-tests.
- **Evidence:**
  - *Academic/Research vs Finance:* Academic/Research higher (M=3.50 vs 2.00); $t(6.9) = 2.88$, $p = 0.024$, $d = 1.76$, 95% CI [0.26, 2.74].
  - *Academic/Research vs Production Planning:* Academic/Research higher (3.50 vs 2.28); $t(17.0) = 3.49$, $p = 0.003$, $d = 1.35$, 95% CI [0.48, 1.96].
  - *Academic/Research vs Non-Technical (pooled):* Academic/Research higher (3.50 vs 2.30); $t(13.6) = 3.76$, $p = 0.002$, $d = 1.31$, 95% CI [0.51, 1.89].

Overall, the pattern of Welch *t*-tests is consistent with the claim that programming is salient in academic roles alongside technical roles.

4.3.2 Research Question 1: Database Knowledge

– **Key Result:** Database knowledge is broadly valued; lack of DB proficiency limits effectiveness not only in technical roles but also in non-technical departments.

– **Measure & test:** Q15 ("Has lack of DB proficiency limited your effectiveness?") was binarized (Yes vs Others). Group differences were evaluated using two-proportion z-tests, reporting p, Cohen's h, and 95% CIs for the difference in Yes rates.

– **Evidence:**

  ∗ *Tech vs Non-Tech (Top-6):* Yes = 57.1% vs 35.1%; diff = 22.0 pp; 95% CI [0.5, 43.5] pp; $z = 1.96$, $p = 0.050$, $h = 0.45$.

  ∗ *Tech vs Non-Tech (All):* Yes = 56.8% vs 29.5%; diff = 27.3 pp; 95% CI [7.4, 47.2] pp; $z = 2.58$, $p = 0.010$, $h = 0.56$.

Across departments, lack of database proficiency limits effectiveness not only in technical roles but also in non-technical ones (Top-6: 57.1% vs 35.1%, $z = 1.96$, $p = 0.050$, $h = 0.45$; All: 56.8% vs 29.5%, $z = 2.58$, $p = 0.010$, $h = 0.56$), indicating that DB competence is broadly valued across the departments.

4.3.3 Research Question 2: Academic Preparation

– **Key Result:** Across departments, undergraduate preparation in programming, OOP, and database concepts was *mostly rated as partially adequate or inadequate*. Very few respondents reported fully sufficient training, and perceived gaps are especially visible in technical roles (e.g., IT). Database-concepts preparation shows a broad shortfall.

- **Measures:** Programming/OOP education adequacy was captured on a 5-point scale (Very adequate $\rightarrow$ Very inadequate) and summarized by department. OOP and Database education sufficiency were summarized on 3-level scales (Yes / Partially / No).
- **Evidence (descriptive):**
  * *Programming & OOP education:* Department-wise bars are dominated by *Partially adequate* and *Inadequate*; *Very adequate* is rare. Gaps appear sharper in technical departments.
  * *Database concepts:* Most departments concentrate in *Partially* or *No*, indicating a widespread shortfall in DB-related academic preparation.
- **Analytical note:** Because RQ2 targets the *prevalence and distribution* of adequacy ratings rather than group differences, *no additional inferential tests were conducted in the main text*. (If confirmation is desired in an appendix, adequacy-by-department can be examined with chi-square tests of independence with Cramér's V; pairwise contrasts such as *Adequate vs. Not* can use two-proportion z-tests/Fisher; ordinal patterns can be modeled via proportional-odds logistic regression.)

4.3.4 Research Question 3: Technical Proficiency and Professional Outcomes

- **Scope:** This question focuses on the *prevalence* of perceived impacts; results are presented descriptively as proportions and counts.
- **Key descriptive findings:**
  * *OOP proficiency gaps (impact on effectiveness):* Yes $\approx$ 16.7%, No $\approx$ 37.8%, Does not apply $\approx$ 45.6%.
  * *Database proficiency gaps (impact on effectiveness):* Yes $\approx$ 42.7%, No

$\approx 31.5\%$, Does not apply $\approx 25.8\%$.

  * *Positive impacts (DB proficiency):* Most frequently cited in problem solving, project success, and team collaboration; very few reported "no noticeable impact."

- **Analytical note:** Because the figures directly reflect respondents' selections and the aim here is descriptive, *no additional inferential tests were conducted for RQ3*.

4.3.5 Research Question 4: Value of Prior Knowledge (Tech vs Non-Tech)

- **Key Result:** Technical roles place somewhat more value on prior programming/database knowledge; effects are directionally consistent but do not reach the $\alpha = 0.05$ threshold.

- **Scale & tests:** Importance was coded 1–5 (Very significant=5 $\rightarrow$ Unsure=1) and compared with independent-samples Welch t-tests. A top-2-box proportion (Very significant/Significant) was also compared with two-proportion z-tests (Fisher when needed). We report t/df/p, Cohen's d, and 95% CIs for mean differences; for proportions, percentage-point (pp) differences with 95% CIs, z/p, and Cohen's h.

- **Evidence:**

  * *Means (Welch t-test; 1–5).* Top-6: TECH $M = 3.45$ ($n = 42$) vs NON-TECH $M = 3.14$ ($n = 37$); $t(77.0) = 1.24$, $p = 0.218$ (ns), $d = 0.28$, 95% CI $[-0.19, 0.83]$.

    All: TECH $M = 3.43$ ($n = 44$) vs $M = 3.25$ ($n = 44$); $t(84.5) = 0.75$, $p = 0.455$ (ns), $d = 0.16$, 95% CI $[-0.30, 0.66]$.

  * *Top-2-box (two-proportion z-test).* All: TECH 48.0% ($n = 50$) vs NON-TECH 33.3% ($n = 54$); diff $= +14.7$ pp, 95% CI $[-4.0, 33.4]$ pp; $z =$

1.52, $p = 0.128$ (ns), $h = 0.30$.

Top-6: TECH 47.9% ($n = 48$) vs 32.6% ($n = 43$); diff $= +15.4$ pp, 95% CI $[-4.5,\ 35.3]$ pp; $z = 1.49$, $p = 0.136$ (ns), $h = 0.31$.

Overall, technical roles report higher perceived importance, but differences are small-to-moderate and not statistically significant.

4.3.6 Research Question 4: Trust in AI Outputs (Tech vs Non-Tech)

– **Key Result:** Technical roles report higher trust in AI outputs (i.e., are less cautious on average). The mean difference is statistically significant in the full sample; Top-6 shows the same direction but is not significant.

– **Scale & tests:** Trust was coded 1–4 (Completely trust=4, Trust=3, Neutral=2, Do not trust=1) and compared with Welch t-tests. A *cautious* indicator (Neutral + Do not trust) was also compared using two-proportion z-tests (Fisher when needed). We report t/df/p, Cohen's d, and 95% CIs; for proportions, pp differences with 95% CIs, z/p, and Cohen's h.

– **Evidence:**

  * *Means (Welch t-test; 1–4, higher = more trust).* All: TECH $M = 2.50$ ($n = 44$) vs NON-TECH $M = 2.20$ ($n = 44$); $t(82.5) = 2.08$, $p = 0.041$, $d = 0.44$, 95% CI $[0.01,\ 0.58]$.

    Top-6: TECH $M = 2.50$ ($n = 42$) vs $M = 2.27$ ($n = 37$); $t(76.6) = 1.51$, $p = 0.134$ (ns), $d = 0.34$, 95% CI $[-0.07,\ 0.53]$.

  * *Cautious proportion (two-proportion z-test; Neutral + Do not trust).* All: TECH 44.0% ($n = 50$) vs NON-TECH 57.4% ($n = 54$); diff $= -13.4$ pp, 95% CI $[-32.5,\ 5.7]$ pp; $z = -1.37$, $p = 0.172$ (ns), $h = -0.27$.

    Top-6: TECH 43.8% ($n = 48$) vs 55.8% ($n = 43$); diff $= -12.1$ pp, 95%

CI $[-32.5,\ 8.4]$ pp; $z = -1.15$, $p = 0.250$ (ns), $h = -0.24$.

Overall, both mean and proportion-based analyses point in the same direction: technical roles exhibit higher trust; the mean difference is statistically significant in the full sample, whereas proportion contrasts are directionally aligned but not significant.

4.4 Instructor Perspective and Curriculum Reflections

Survey results offer a data-driven view of technical preparedness among industrial and systems engineers, but classroom experience provides a complementary lens. Over four semesters, the second author taught the compulsory "Programming for Operations Research" course to nearly 300 ISE students. The course used Java and treated object-oriented design not merely as a way to write code, but as a framework for structuring and solving problems through abstraction, modularity, and decomposition—skills fundamental to ISE.

The department later migrated the course to a more accessible scripting language and assigned it to a different instructor. Student comfort and participation increased, yet conceptual depth declined. The issue was less the language itself Python, for example, fully supports object-oriented principles—than the way it was typically taught and used: as a quick route to immediate results rather than a disciplined approach to building larger systems. As a result, students often became adept at syntax and libraries but had fewer opportunities to practice decomposition, architectural design, and system-level abstraction.

This teaching experience aligns with the survey evidence. Although many respondents reported intermediate or advanced programming proficiency (Figure 4.1), far fewer felt that their university education prepared them to apply OOP concepts in practice (Figure 4.8). Moreover, familiarity with foundational ideas such as the SOLID principles was limited, suggesting that instruction often remains superficial or fragmented. Overall, a consistent pattern emerges: programming is increasingly present in ISE curricula, yet it is frequently taught without the depth, structure, or purpose required by industry.

From an instructional standpoint, curricula should emphasize algorithmic thinking and system-level design alongside coding. Scripting languages can ease entry, but programs must preserve the habits of abstraction and analytical reasoning that define ISE. Otherwise, there is a risk that programming will function as a black box, used by future engineers but not fully understood.

5. CONCLUSION

This study has investigated the role of programming proficiency (particularly OOP), database knowledge, and the integration of AI-based tools in the professional practices of Industrial and Systems Engineers. By examining the perceptions and experiences of professionals across various sectors, the research aimed to assess whether current educational curricula align with practical industry requirements and how technical competencies contribute to engineers' career development and effectiveness.

The findings indicate that while OOP and database skills are not universally essential across all engineering roles, they contribute meaningfully to enhancing problem-solving capabilities, project success. Importantly, the results suggest that database knowledge may be perceived as somewhat more essential than OOP skills across diverse engineering contexts, reflecting its broader applicability in data-driven tasks and system integration. However, the high proportion of respondents indicating uncertainty or limited applicability underscores that the relevance of these technical skills is strongly context-dependent and varies according to sector-specific demands and individual roles. Moreover, the study highlights a cautious yet growing engagement with AI-based tools among engineers. Although many professionals regularly employ these tools for learning, research, and technical assistance, there remains a moderate level of trust in their outputs, particularly in domains where precision and domain-specific expertise are critical. The perceived necessity of prior programming or database knowledge for effectively using AI tools also varies across departments, reflecting diverse technical expectations and workplace practices.

These insights underline the importance of continuously adapting ISE curricula to keep pace with emerging technologies and evolving industry expectations. While not every role demands advanced programming expertise, equipping students with foundational knowledge in OOP, database management so analytical thinking, and AI tool literacy can enhance their flexibility and problem-solving potential in a dynamic professional landscape. Based on these findings, the following curriculum enhancements are recommended to better prepare future Industrial and Systems Engineers for the evolving demands of the profession:

1. Integrating OOP concepts into core courses such as OR, statistics, and simulation to reinforce their practical applicability and relevance.

2. Adding database-related courses to the curriculum to equip students with comprehensive data handling and analytical skills, enabling them to work effectively with data and develop data-driven solutions.

3. Incorporating training on AI-based tools alongside programming skills to enhance adaptability and effective tool utilization. This approach also provides an opportunity to familiarize undergraduate students with the ethical use of AI technologies, which is increasingly important in today's professional environment.

A notable limitation of this study lies in its sample composition, which predominantly consists of participants from Turkey, USA, and Iran potentially limiting the generalizability of the findings to broader international contexts. Future research could expand this scope by engaging a more diverse and larger cohort of participants from multiple countries and industrial sectors. Such studies would provide richer insights into global variations in technical skill requirements and contribute to developing more universally relevant educational strategies for Industrial and Systems Engineers.

Overall, the insights from this study not only clarify the significance of programming, database knowledge, and AI competencies in ISE but also chart a path toward educational reforms that can better equip future engineers to navigate and lead in an era of rapid technological transformation.

Ultimately, the real question is not whether Industrial and Systems Engineers can use modern tools—like large language models and AI assistants—to write code or query databases. They certainly can, and many already do so in practice. In today's world, where syntax, code snippets, and database queries are just a search away, simply knowing how to produce code is no longer what sets an engineer apart. The more important issue is whether future engineers are being trained to handle complexity: to design scalable systems, ensure data integrity, understand process dependencies, and know why a database should be normalized or why software modules should be decoupled—not just how to ask ChatGPT for a solution. Without developing this kind of analytical depth, we risk producing Industrial and Systems Engineers who can automate but not truly analyze, who can deploy solutions but not design them thoughtfully. Engineering education needs to prepare students not only to use modern tools effectively but also to think critically about the underlying systems and principles behind those tools.

REFERENCES

[1] A. Sharma, F. Khan, D. Sharma, S. Gupta, and F. Student, “Python: the programming language of future,” *Int. J. Innovative Res. Technol*, vol. 6, no. 2, pp. 115–118, 2020.

[2] A. Saabith, M. Fareez, and T. Vinothraj, “Python current trend applications-an overview,” *International Journal of Advance Engineering and Research Development*, vol. 6, no. 10, 2019.

[3] University of Florida Industrial & Systems Engineering, “Model Semester Plan,” 2025. [Accessed July 2025].

[4] Temple University, College of Engineering, “Industrial and systems engineering major - required courses,” 2025. Accessed: 2025-07-08.

[5] Virginia Tech, Grado Department of Industrial and Systems Engineering, “Ise undergraduate course descriptions,” 2025. Accessed: 2025-07-08.

[6] Auburn University, Department of Industrial and Systems Engineering, “Ise undergraduate curriculum,” 2025. Accessed: 2025-07-08.

[7] M. S. Sodhi and B.-G. Son, “Industry requirements of operations research skills based on statistical content analysis of job ads,” *Available at SSRN 1011468*, 2007.

[8] H. Eskandari, S. Sala-Diakanda, S. Furterer, L. Rabelo, L. Crumpton-Young, and K. Williams, “Enhancing the undergraduate industrial engineering curriculum: Defining desired characteristics and emerging topics,” *Education+ Training*, vol. 49, no. 1, pp. 45–55, 2007.

[9] B. L. Treviño-Elizondo and H. García-Reyes, “What does industry 4.0 mean to industrial engineering education?,” *Procedia Computer Science*, vol. 217, pp. 876–885, 2023.

[10] R. Pérez-Rodríguez, R. Lorenzo-Martin, C. A. Trinchet-Varela, R. E. Simeón-Monet, J. Miranda, D. Cortés, and A. Molina, “Integrating challenge-based-learning, project-based-learning, and computer-aided technologies into industrial engineering teaching: Towards a sustainable development framework,” *Integration of Education*, vol. 26, no. 2, pp. 198–215, 2022.

[11] J. Kádárová, J. Kováč, M. Durkáčová, and G. Kádár, “Education in industrial engineering in slovakia,” *Procedia-Social and Behavioral Sciences*, vol. 143, pp. 157–162, 2014.

[12] J. Mula and M. Bogataj, “Or in the industrial engineering of industry 4.0: experiences from the iberian peninsula mirrored in cjor,” *Central European Journal of Operations Research*, vol. 29, pp. 1163–1184, 2021.

[13] J. R. Hardin, A. Holder, J. C. Beck, K. Furman, A. Hanna, D. Rader, and C. Rego, “Recommendations for an undergraduate curriculum at the interface of operations research and computer science,” *INFORMS Transactions on Education*, vol. 12, no. 3, pp. 117–123, 2012.

[14] M. S. Sodhi, B.-G. Son, and C. S. Tang, “Asp, the art and science of practice: what employers demand from applicants for mba-level supply chain jobs and the coverage of supply chain topics in mba courses,” *Interfaces*, vol. 38, no. 6, pp. 469–484, 2008.

[15] A. Gilson, C. W. Safranek, T. Huang, V. Socrates, L. Chi, R. A. Taylor, D. Chartash, *et al.*, “How does chatgpt perform on the united states medical licensing examination (usmle)? the implications of large language models for medical education and knowledge assessment,” *JMIR medical education*, vol. 9, no. 1, p. e45312, 2023.

[16] K. Fuchs, “Exploring the opportunities and challenges of nlp models in higher education: is chat gpt a blessing or a curse?,” in *Frontiers in education*, vol. 8, p. 1166682, Frontiers Media SA, 2023.

[17] H. Yu, “Reflection on whether chat gpt should be banned by academia from the perspective of education and teaching,” *Frontiers in Psychology*, vol. 14, p. 1181712, 2023.

[18] M. M. Rahman and Y. Watanobe, “Chatgpt for education and research: Opportunities, threats, and strategies,” *Applied sciences*, vol. 13, no. 9, p. 5783, 2023.

[19] A. Azaria, “ChatGPT usage and limitations,” tech. rep., Ariel University, 2022. DOI: 10.31219/osf.io/5ue7n.

[20] D. Baidoo-Anu and L. O. Ansah, “Education in the era of generative artificial intelligence (ai): Understanding the potential benefits of chatgpt in promoting teaching and learning,” *Journal of AI*, vol. 7, no. 1, pp. 52–62, 2023.

[21] E. L. Ouh, B. K. S. Gan, K. Jin Shim, and S. Wlodkowski, “Chatgpt, can you generate solutions for my coding exercises? an evaluation on its effectiveness in an undergraduate java programming course.,” in *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1*, pp. 54–60, 2023.

[22] R. Khojah, M. Mohamad, P. Leitner, and F. G. de Oliveira Neto, “Beyond code generation: An observational study of chatgpt usage in software engineering practice,” *Proceedings of the ACM on Software Engineering*, vol. 1, no. FSE, pp. 1819–1840, 2024.

APPENDICES

APPENDIX A. SURVEY QUESTIONNAIRE

The following survey questions have been designed to gather demographic information and assess the participants' knowledge and experience in various technical areas. These questions aim to understand the background of the participants, their expertise, and their interaction with tools such as programming languages, databases, and AI-based technologies. The first part of the survey focuses on collecting basic demographic details to better contextualize the responses and provide insights into the diversity of the participants.

A.1 Part 0: Demographic Information

The following questions aim to gather demographic information about the participants' educational background, professional experience, and current work environment. This section is designed to help better understand the context and profiles of the participants.

In what year did you complete your undergraduate degree?

Where did you complete your undergraduate degree?

- USA
- Canada
- UK
- Turkey
- Australia
- Germany
- China
- Brazil
- Other (Please specify)

Which of the following best describes your undergraduate degree?

- Industrial Engineering
- Mechanical Engineering
- Electrical Engineering

- Computer Science
- Business Administration
- Mathematics
- Other (Please specify)

What is your highest degree and major?

- Bachelor's in Industrial Engineering
- Bachelor's in Computer Engineering
- Master's in Industrial Engineering
- Master's in Computer Engineering
- PhD in Industrial Engineering
- PhD in Computer Engineering
- Other (Please specify)

In which sector are you currently working?

- IT
- Retail
- Tech
- Finance
- Production
- Marketing
- Mining
- Academic
- Consulting
- Other (Please specify)

How many years of professional experience do you have?

- 1-3 years
- 4-8 years

- 8-14 years
- 15+ years

What is your current job position or title?

In which country are you currently employed?

Which department do you work in?

- IT
- Production Operations Planning
- Marketing
- R&D
- Other (Please specify)

A.2 Part 1: Programming Knowledge and Experience

This section focuses on the participants' programming knowledge, the languages they are familiar with, and how they utilize these skills in their professional work. The following questions will help to understand the programming proficiency and experience of the participants.

1. How would you rate your programming knowledge?

- Novice
- Beginner (understands basic syntax, data structures, and flow control, e.g., loops, conditionals)
- Intermediate
- Advanced (capable of implementing complex algorithms)

2. Which programming or querying languages are you proficient in? Please select all that apply.

- Python
- C#
- C
- C++

- Java
- Go
- Rust
- Ruby
- JavaScript
- Julia
- SQL
- Other (Please specify)

3. What resources did you use to learn programming? Please select all that apply.

- Online courses
- University classes
- Self-taught
- Other (Please specify)

4. How frequently do you use programming languages in your work?

- Daily
- Weekly
- Monthly
- Rarely
- Never

5. How critical is programming knowledge for your role?

- Very Important
- Important
- Moderate
- Slight
- Not Important

6. How adequate was the programming knowledge you gained in university for your professional life?

- Very Adequate
- Adequate
- Partially adequate
- Inadequate
- Very inadequate

A.3 Part 2: Object-Oriented Programming (OOP) Knowledge and Experience

This section evaluates the participants' knowledge of Object-Oriented Programming (OOP) and how these concepts are applied in their professional lives. The questions aim to explore familiarity with OOP principles, real-world application, and the impact of OOP knowledge on career development.

7. How would you rate your knowledge of Object-Oriented Programming concepts?

- Novice
- Beginner (knows basic syntax, class structure, and can create simple programs)
- Intermediate (understands object references, encapsulation, and inheritance)
- Advanced (can develop complex class hierarchies and design patterns)

8. Have you heard about the SOLID (Single responsibility, Open-closed, Liskov substitution, Interface segregation, Dependency inversion) principles?

- Yes
- No

9. Was your university education sufficient for applying OOP in real-world industry problems?

- Yes
- Partially
- No

10. Have you encountered situations where the lack of proficiency in OOP concepts limited your effectiveness?

- Yes
- No
- Uncertain/Does not apply

11. Which aspects of your career have been positively influenced by proficiency in OOP concepts? Please select all that apply.

- Promotion opportunities
- Project success
- Team collaboration
- Problem-solving
- No noticeable impact
- Uncertain/Does not apply

A.4 Part 3: Database Knowledge and Experience

This section focuses on the participants' experience and proficiency with advanced programming topics and technologies. It aims to assess their familiarity with concepts like algorithms, data structures, and emerging technologies in the field of programming.

12. How would you rate your proficiency in implementing algorithms and data structures?

- Novice
- Beginner (can implement basic data structures like arrays and linked lists)
- Intermediate (can implement and use algorithms such as sorting and searching)
- Advanced (can implement complex algorithms like graph algorithms and dynamic programming)

13. Which of the following technologies are you familiar with? Please select all that apply.

- Cloud Computing (AWS, Azure, Google Cloud)
- Machine Learning (TensorFlow, Scikit-learn, Keras)

- Big Data Technologies (Hadoop, Spark, Kafka)
- Containers (Docker, Kubernetes)
- DevOps Tools (Jenkins, GitLab CI/CD, Ansible)
- Blockchain
- Other (Please specify)

14. Have you worked with version control systems such as Git?

- Yes
- No

15. How often do you apply advanced programming techniques in your work?

- Daily
- Weekly
- Monthly
- Rarely
- Never

16. How would you rate your knowledge of testing and debugging techniques?

- Novice
- Beginner (can debug simple errors and write basic unit tests)
- Intermediate (can use debugging tools and write comprehensive tests)
- Advanced (can design and implement advanced testing strategies and debugging techniques)

A.5 Part 4: AI-Based Tools Utilization

This section explores how participants utilize AI-based tools like ChatGPT in their work, including the frequency of use, the purposes for which they use them, and their trust in these tools. The aim is to understand the role of AI tools in enhancing work processes.

17. How often do you use ChatGPT or similar AI-based tools?

- Daily
- Weekly
- Monthly
- Rarely
- Never

18. For what purposes do you use ChatGPT or similar AI tools? Please select all that apply.

- Programming assistance
- Problem solving
- Learning and research
- Content creation
- I do not utilize these tools
- Other (Please specify)

19. How much do you trust the outputs provided by ChatGPT or similar AI-based tools?

- Completely trust
- Trust
- Neutral
- Do not trust

20. How would you rate your experience with AI Tools?

- Very satisfying
- Satisfying
- Neutral

- Unsatisfying

21. How significant do you believe prior knowledge in programming or databases is for effectively utilizing ChatGPT or similar AI tools?

- Very significant
- Significant
- Somewhat significant
- Not significant
- Unsure

APPENDIX B. DEMOGRAPHIC CHARTS (Q1–Q9)

Figure B.1: *Graduation Year Distribution*

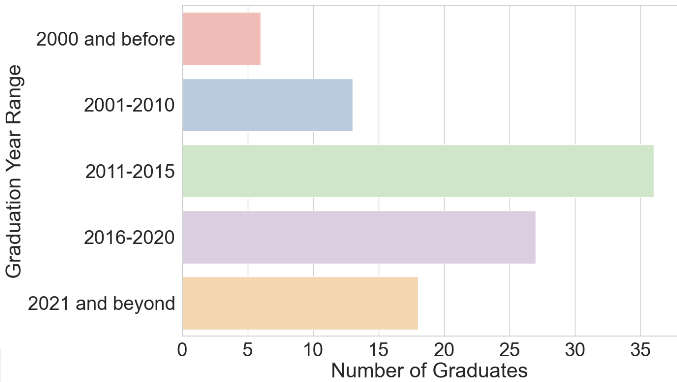


Figure B.2: *Undergraduate Degree Countries Distribution*

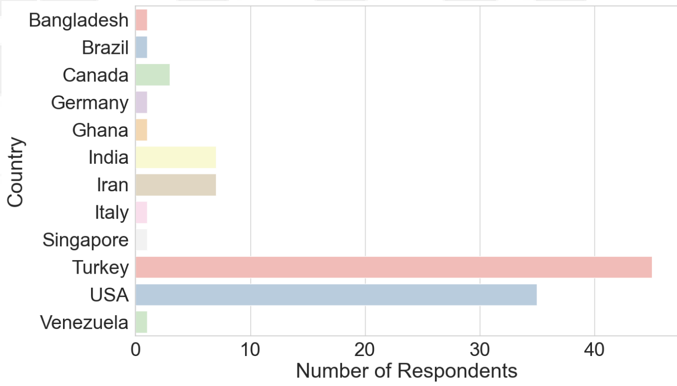


Figure B.3: *Undergraduate Degree Types Distribution*

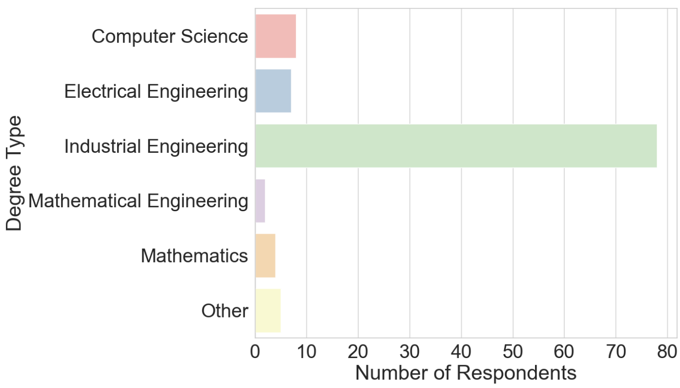


Figure B.4: *Highest Degree and Major Distribution*

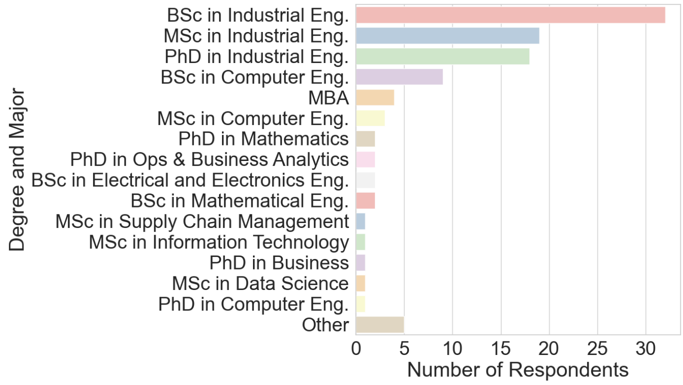


Figure B.5: *Sector Distribution*

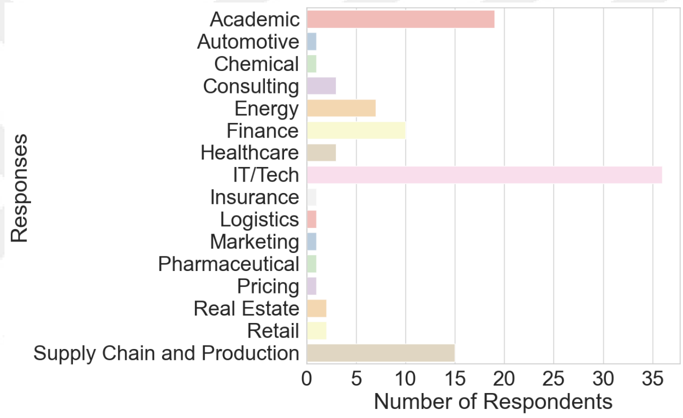


Figure B.6: *Experience Level Distribution*

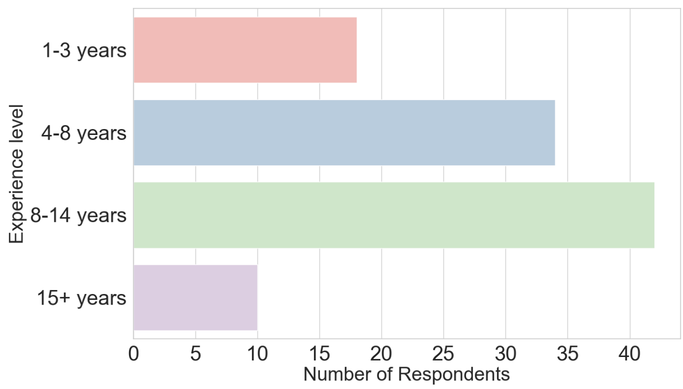


Figure B.7: *Job Title Categories Distribution*

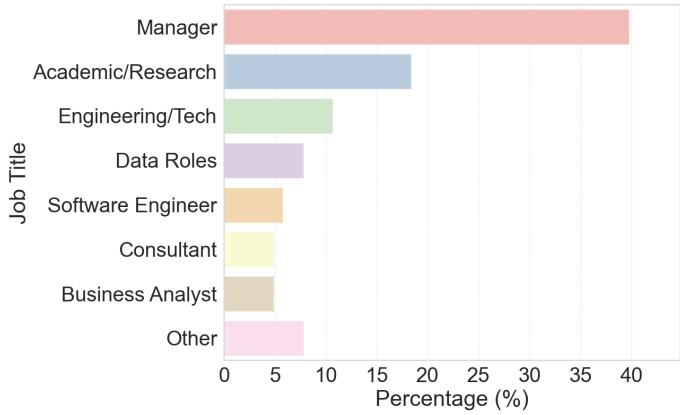


Figure B.8: *Country of Employment Distribution*

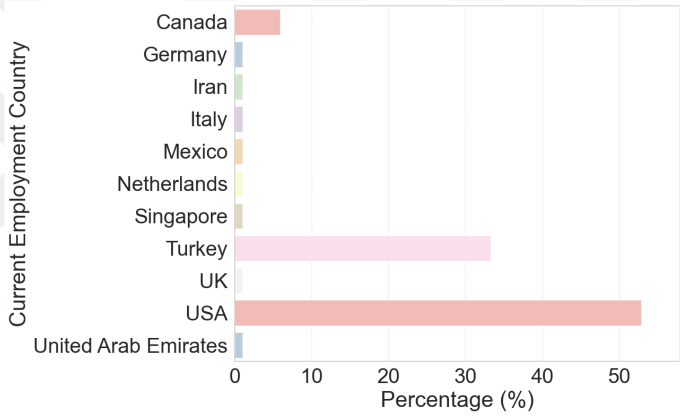


Figure B.9: *Department Distribution*