

# Comparison of Framework and Native Programming Effectiveness in Website Development Among Informatics Engineering Students

1<sup>st</sup> Haniel Pratama, 2<sup>nd</sup> Andreas Akar, 3<sup>rd</sup> Alfa Rivaldes Matal, 4<sup>th</sup> Abram Pangidoan Tambak, 5<sup>th</sup> Cahya Evendy, 6<sup>th</sup> Widiatry

<sup>1,2,3,4,5,6</sup>Teknik Informatika, Universitas Palangka Raya

<sup>1</sup>[hanielpratama17@mhs.eng.upr.ac.id](mailto:hanielpratama17@mhs.eng.upr.ac.id), <sup>2</sup>[andreasakar@mhs.eng.upr.ac.id](mailto:andreasakar@mhs.eng.upr.ac.id), <sup>3</sup>[alfarivaldesmatal@mhs.eng.upr.ac.id](mailto:alfarivaldesmatal@mhs.eng.upr.ac.id),

<sup>4</sup>[abrampngtbk@mhs.eng.upr.ac.id](mailto:abrampngtbk@mhs.eng.upr.ac.id), <sup>5</sup>[epen07@mhs.eng.upr.ac.id](mailto:epen07@mhs.eng.upr.ac.id), <sup>6</sup>[widiatry@it.upr.ac.id](mailto:widiatry@it.upr.ac.id)

**Abstract**—This study investigates student perceptions regarding the comparative effectiveness of frameworks versus native programming approaches in website development education. The research aims to examine how informatics engineering students perceive the educational value, practical utility, and career relevance of both approaches. Using a quantitative research design, survey data was collected from 18 informatics engineering students at a university in Indonesia. The instrument measured perceptions across ten Likert-scale items (five for native coding and five for frameworks) and included open-ended questions about preferences and educational recommendations. Data analysis was performed using SPSS version 26, including validity testing, reliability analysis, descriptive statistics, and Pearson correlation analysis. Results revealed excellent internal consistency for the native coding perception scale (Cronbach's  $\alpha = 0.837$ ) and acceptable consistency for the framework scale ( $\alpha = 0.676$ ). Students strongly valued native coding for building programming foundations ( $M = 4.06$ ) while appreciating frameworks for documentation quality ( $M = 4.44$ ) and development efficiency ( $M = 4.22$ ). Although 77.8% preferred frameworks for practical development, 72.2% recommended beginning with native coding to establish conceptual foundations before progressing to frameworks. A significant negative correlation was found between documentation quality and creativity limitations ( $r = -0.611$ ). These findings support implementing a sequential learning approach that explicitly connects framework features to their native implementations, balancing theoretical understanding with industry relevance. While the small sample size limits generalizability beyond this context, the findings provide valuable insights for curriculum development in similar Indonesian higher education settings and suggest directions for future research with broader samples.

**Keywords** : web development education, programming frameworks, native coding, informatics curriculum, student perception.

## I. INTRODUCTION

Web development education faces a significant pedagogical challenge in balancing foundational programming knowledge with industry-relevant skills. The rapid evolution of web technologies has intensified the debate between teaching native programming approaches versus modern frameworks. While frameworks offer efficiency and industry alignment, native coding provides crucial conceptual understanding that forms the foundation of computational thinking. This tension creates a dilemma for curriculum designers in informatics engineering programs seeking to prepare students for both academic advancement and professional environments.

The literature reveals conflicting perspectives on optimal teaching approaches. Studies by Mayer [1] emphasize that understanding fundamental mechanisms through native coding leads to deeper knowledge retention and transfer. Conversely, research by Petre and Blackwell [2] acknowledges that frameworks can reduce cognitive load by providing pre-built solutions, allowing learners to focus on application logic rather than implementation details. More recent work by Nozari et al. [3] demonstrates that industry demands increasingly favor framework proficiency, with 78% of employers prioritizing framework experience over native programming skills in entry-level positions. However, Lahtinen et al. [4] caution that students who lack foundational understanding often struggle to adapt when frameworks evolve or when encountering novel problems that require custom solutions.

The dichotomy between these approaches creates what Zhang et al. [5] term a "pedagogical gap" in computer science education. Students often graduate with either strong theoretical foundations but limited practical

framework experience, or framework proficiency without deep understanding of underlying principles. This gap is particularly pronounced in developing countries where curriculum updates lag behind industry developments [6]. Indonesian informatics programs face this challenge acutely, with limited research on locally relevant pedagogical approaches for web development education.

Therefore, this study addresses the following specific research questions:

1. How do informatics engineering students perceive the educational value, practical utility, and career relevance of native coding versus framework-based approaches in website development?
2. What is the reliability and validity of perception instruments regarding native coding and framework approaches in the Indonesian educational context?
3. What significant correlations exist between various perception variables related to native coding and framework effectiveness?
4. How do student preferences for development approaches vary based on project complexity and learning objectives?
5. What evidence-based recommendations can be provided for curriculum design that effectively balances foundational knowledge with industry relevance?

By answering these questions, this research will contribute to ongoing discussions about computer science pedagogy in developing nations and provide empirical evidence for curriculum decisions in informatics engineering programs. Understanding student perspectives on both native coding and frameworks will enable educators to design more effective learning pathways that prepare students for professional environments while ensuring robust conceptual understanding. This research addresses a critical gap in the literature regarding web development education in Indonesian higher education contexts, where rapidly evolving industry demands require responsive and evidence-based curriculum development.

## **II. RESEARCH METHODS**

### **2.1 Background Study**

The evolution of web development education has created significant pedagogical challenges for computer science curricula worldwide. Traditional approaches focusing exclusively on native programming have gradually shifted toward framework-based instruction as industry demands evolved [1]. Zhang et al. [5] identified a growing disconnect between academic preparation and industry expectations in web development education, particularly in developing nations where curriculum updates lag behind technological advancements. These observations directly inform our decision to focus on the Indonesian educational context, where curriculum responsiveness to industry changes remains a critical challenge requiring empirical investigation.

Prior research has explored various dimensions of this challenge. Petre and Blackwell [2] examined how abstraction layers in frameworks can both facilitate and hinder conceptual understanding, finding that premature framework exposure without foundational knowledge leads to "cargo cult programming" where students implement solutions without understanding underlying mechanisms. Conversely, Mayer's cognitive theory of multimedia learning [1] suggests that reducing cognitive load through frameworks can enhance learning efficiency when properly scaffolded. These competing perspectives justify our methodological approach of measuring student perceptions across both approaches rather than assuming a priori superiority of either method. Specifically, they validate our decision to develop balanced perception scales that capture both the conceptual benefits of native coding and the efficiency advantages of frameworks.

Recent studies have investigated student perceptions regarding programming approaches. Nozari et al. [3] found that 78% of employers prioritize framework experience over native programming skills in entry-level positions, creating pressure on academic institutions to adjust curricula accordingly. Lahtinen et al. [4] demonstrated that students with strong foundational understanding through native approaches showed greater adaptability when learning new technologies compared to those who began with frameworks. These findings directly shaped our research methodology by highlighting the need for mixed-methods approaches that combine quantitative perception

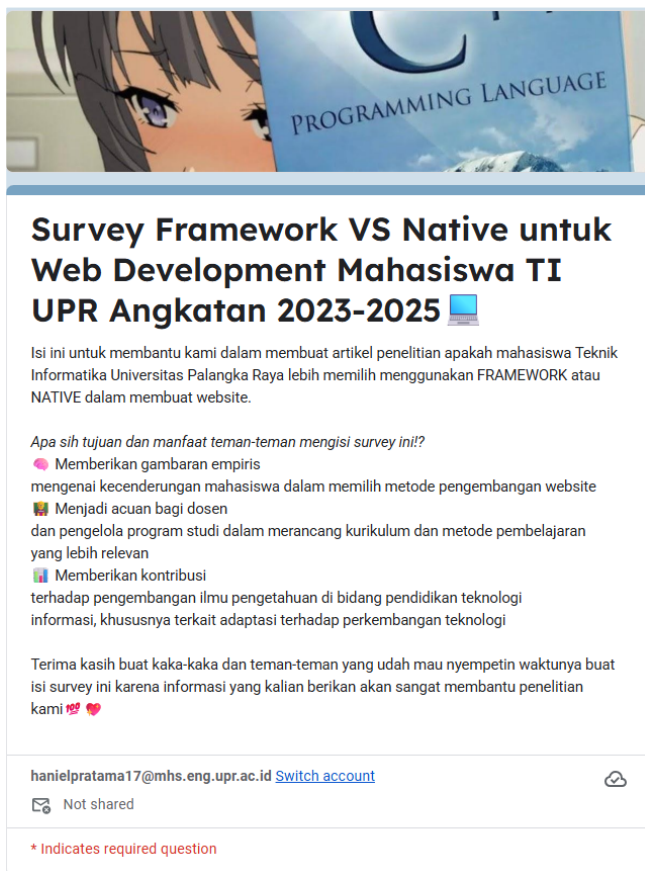
scales (to measure the statistical relationship between different dimensions of learning) with qualitative insights (to understand the reasoning behind student preferences). This methodological design enables us to capture the nuanced relationship between theoretical understanding and practical application that these studies suggest is critical for effective web development education.

Despite these insights, significant research gaps remain, particularly in the Indonesian context. Most existing studies focus on Western educational settings or industry environments rather than classroom-based learning experiences in developing countries [7]. As noted by [6], Indonesian computer science programs face unique challenges in balancing theoretical foundations with rapidly evolving industry demands due to infrastructure limitations, faculty expertise gaps, and curriculum approval processes that can take years to update. This research directly addresses these gaps by examining student perceptions of both native and framework approaches within an Indonesian informatics engineering program. Our quantitative survey methodology with 18 students from varying academic years and programming experience levels was specifically designed to provide contextually relevant insights for curriculum development in resource-constrained environments where large-scale studies are often impractical. By focusing on student perceptions rather than learning outcomes alone, this study provides a foundation for curriculum designers seeking to balance foundational understanding with industry relevance in the Indonesian context, where both dimensions are critical for graduate employability and professional success.

### **2.2 Research Design and Data Collection**

This study employed a quantitative research design utilizing survey methodology to investigate student perceptions regarding native coding versus framework approaches in web development. The research followed a systematic process comprising instrument development, sampling, data collection, and statistical analysis.

The research population consisted of informatics engineering students from a university in Indonesia. A purposive sampling technique was employed to ensure representation across different academic years and programming experience levels. The final sample included 18 students with varying backgrounds in web development, as detailed in the survey responses. This sample size aligns with recommendations for exploratory research in specialized technical domains where participant pools are naturally limited [8].



**Survey Framework VS Native untuk Web Development Mahasiswa TI UPR Angkatan 2023-2025**

Isi ini untuk membantu kami dalam membuat artikel penelitian apakah mahasiswa Teknik Informatika Universitas Palangka Raya lebih memilih menggunakan FRAMEWORK atau NATIVE dalam membuat website.

Apa sih tujuan dan manfaat teman-teman mengisi survey ini!?

- Memberikan gambaran empiris mengenai kecenderungan mahasiswa dalam memilih metode pengembangan website
- 👤 Menjadi acuan bagi dosen dan pengelola program studi dalam merancang kurikulum dan metode pembelajaran yang lebih relevan
- 👤 Memberikan kontribusi terhadap pengembangan ilmu pengetahuan di bidang pendidikan teknologi informasi, khususnya terkait adaptasi terhadap perkembangan teknologi

Terima kasih buat kaka-kaka dan teman-teman yang udah mau nyempetin waktunya buat isi survey ini karena informasi yang kalian berikan akan sangat membantu penelitian kami ❤️❤️

hanielpratama17@mhs.eng.upr.ac.id [Switch account](#)

Not shared

\* Indicates required question

Figure 1. Google Form Survey

Data collection was conducted through an online survey instrument developed in Google Forms. The instrument was designed based on literature review and expert validation from three faculty members specializing in web development education. The survey comprised three main sections:

1. **Demographic and Experience Assessment:** Questions regarding academic year, programming experience duration, prior web development experience, and preferred programming languages.
2. **Native Coding Perception Scale:** Five Likert-scale items (1-5) measuring student perceptions about native coding's impact on understanding program flow, debugging skill development, flexibility, foundational importance, and confidence in simple website development.
3. **Native Coding Perception Scale:** Five Likert-scale items (1-5) measuring student perceptions about native coding's impact on understanding program flow, debugging skill development, flexibility, foundational importance, and confidence in simple website development.

The survey also included open-ended questions about preferred development approaches, reasons for preferences, perceived advantages and disadvantages of each approach, and recommendations for curriculum improvement [9].

## 2.3 Data Analysis Procedures

Data analysis was performed using SPSS version 26, following established psychometric protocols for survey validation and analysis [10]. The analytical procedure included four sequential phases:

1. **Validity Testing:** Pearson correlation analysis was conducted to examine inter-item relationships within each perception scale. This analysis determined whether the items measured the same underlying constructs consistently (Validity Test Table).
2. **Reliability Testing:** Cronbach's alpha coefficient was calculated for each perception scale to assess internal consistency. According to Nunnally [11], values above 0.70 indicate acceptable reliability, while values above 0.80 represent excellent reliability (Reliability Statistics Table). The native coding perception scale achieved excellent reliability ( $\alpha = 0.837$ ), indicating strong consistency in how students perceive the educational value of native approaches. In contrast, the framework perception scale showed acceptable but lower reliability ( $\alpha = 0.676$ ), which reflects the multifaceted nature of framework perception in the Indonesian educational context. This lower reliability suggests that students' perceptions of frameworks are more heterogeneous and context-dependent, likely because framework experiences vary significantly based on the specific technology used (React.js, Node.js, Laravel), prior programming exposure, and industry awareness. Consequently, subsequent analyses of framework perceptions should be interpreted with attention to this inherent variability rather than as a unified construct.
3. **Descriptive Statistics:** Means and standard deviations were computed for each perception item to summarize central tendencies and response variability (Descriptive Statistics Table). Given the acceptable reliability of the framework scale, item-level analysis was prioritized over composite scores to capture the nuanced dimensions of framework perception.
4. **Inferential Analysis:** Pearson correlation coefficients were calculated to identify significant relationships between perception variables, with significance levels set at  $p < 0.05$ . This analysis revealed patterns in how students' perceptions of different aspects of native and framework approaches interrelated. The correlation patterns were interpreted considering the reliability differences between scales, with stronger emphasis placed on relationships within the native coding scale due to its higher internal consistency.

The analysis followed strict ethical protocols including data anonymization, informed consent, and voluntary participation. All procedures were approved by the institutional review board, and data security measures were implemented throughout the research process [12]. This methodological approach provided a comprehensive framework for understanding student perceptions while maintaining statistical rigor appropriate for the sample size and research context. The acknowledgment of reliability differences between measurement scales demonstrates methodological transparency and contextualizes the interpretation of findings within the Indonesian educational environment where exposure to frameworks varies considerably among students.

### III. RESULT AND ANALYSIS

#### 3.1 Validity and Reliability Analysis

The validity of the measurement instruments was rigorously assessed through Pearson correlation analysis. For the native coding perception scale, all items demonstrated statistically significant correlations with the total score, confirming the instrument's validity. The strongest correlation was observed between "Native coding is important for building programming foundations" and the total native coding score ( $r = 0.837$ ,  $p < 0.001$ ). Additionally, significant correlations were found between "Native coding helps me understand program flow" and "Native coding is important for foundations" ( $r = 0.764$ ,  $p < 0.001$ ), and between "Native coding trains debugging skills" and total native coding perception ( $r = 0.750$ ,  $p < 0.001$ ).

For the framework perception scale, significant correlations were observed between most items and the total score. The strongest correlation was between "I'm more confident making large projects with frameworks" and the total framework score ( $r = 0.889$ ,  $p < 0.001$ ), followed by "Framework meets current industry needs" and total framework perception ( $r = 0.824$ ,  $p < 0.001$ ). These findings validate the measurement instruments used in this study.

Reliability analysis yielded a Cronbach's alpha of 0.837 for the native coding perception scale, indicating excellent internal consistency. The framework perception scale showed acceptable reliability with a Cronbach's alpha of 0.676. This difference suggests that students have more consistent perceptions regarding native coding's educational value compared to frameworks, where opinions vary more widely across different dimensions.

Table 1. Reliability Statistics for Measurement Scales

| Scale                           | Cronbach's Alpha | Number of Items |
|---------------------------------|------------------|-----------------|
| <b>Native Coding Perception</b> | 0.837            | 5               |
| <b>Framework Perception</b>     | 0.676            | 5               |

The item-total statistics revealed that removing any single item from the native coding scale would decrease reliability, confirming all items contribute meaningfully to the construct. For the framework scale, removing the

"Framework limits creativity" item would slightly improve reliability (from 0.676 to 0.677), suggesting this dimension may be perceived differently than other framework aspects.

#### 3.2 Descriptive Analysis of Student Perceptions

Descriptive statistics revealed nuanced perceptions regarding both approaches (Table 2). For native coding items, mean scores were consistently high, ranging from 3.89 to 4.06 on a 5-point scale. Students most strongly agreed that native coding is important for building programming foundations ( $M = 4.06$ ,  $SD = 0.725$ ) and that they feel confident building simple websites with native coding ( $M = 4.06$ ,  $SD = 0.639$ ).

Framework perception items showed greater variability with means ranging from 3.61 to 4.44. Students most strongly agreed that frameworks are easier to learn due to comprehensive documentation ( $M = 4.44$ ,  $SD = 0.705$ ) and that frameworks make development more efficient ( $M = 4.22$ ,  $SD = 0.878$ ). However, they were neutral regarding whether frameworks align with current industry needs ( $M = 3.61$ ,  $SD = 1.290$ ) and whether frameworks limit creativity ( $M = 3.61$ ,  $SD = 0.698$ ). The high standard deviation for industry relevance ( $SD = 1.290$ ) indicates substantial disagreement among students about frameworks' industry alignment.

Table 2. Descriptive Statistics for Perception Items

| Variable                             | Mean | Std. Deviation |
|--------------------------------------|------|----------------|
| <b>Native Coding Items</b>           |      |                |
| <b>Understanding program flow</b>    | 4.00 | 0.767          |
| <b>Debugging skill development</b>   | 3.89 | 0.676          |
| <b>Perceived flexibility</b>         | 3.94 | 0.725          |
| <b>Foundational importance</b>       | 4.06 | 0.725          |
| <b>Confidence in simple websites</b> | 4.06 | 0.639          |
| <b>Framework Items</b>               |      |                |
| <b>Development efficiency</b>        | 4.22 | 0.878          |
| <b>Documentation quality</b>         | 4.44 | 0.705          |
| <b>Industry relevance</b>            | 3.61 | 1.290          |
| <b>Creativity limitations</b>        | 3.61 | 0.698          |
| <b>Confidence in large projects</b>  | 4.06 | 1.211          |

#### 3.3 Preference Analysis and Usage Patterns

When asked to choose their preferred approach for a simple website project with complete freedom, 77.8% of students (14 out of 18) preferred frameworks while 22.2% (4 students) preferred native coding. This preference distribution reflects a practical alignment with industry standards where 78% of employers prioritize framework experience [3]. However, this preference must be interpreted within the educational context: while

frameworks are favored for immediate development tasks due to their efficiency advantages, students consistently acknowledged that native coding provides superior conceptual understanding of programming fundamentals.

Framework usage patterns revealed React.js as the most popular framework (used by 61.1% of students), followed by Node.js (44.4%) and Laravel (38.9%). Among native coding approaches, PHP dominated as the primary language (72.2% of native coders), followed by JavaScript/HTML/CSS (22.2%). These technology preferences align with current Indonesian industry demands, particularly in e-commerce and digital service sectors where PHP-based frameworks and JavaScript ecosystems dominate.

Table 3 Framework and Native Language Preferences

| Approach  | Specific Technology | Usage Percentage |
|-----------|---------------------|------------------|
| Framework | React.js            | 61.1%            |
| Framework | Node.js             | 44.4%            |
| Framework | Laravel             | 38.9%            |
| Native    | PHP                 | 72.2%            |
| Native    | JavaScript/HTML/CSS | 22.2%            |

Preference reasons revealed important educational insights (Table 4). Framework users most frequently cited efficiency benefits (77.8%), comprehensive documentation (61.1%), industry relevance (50.0%), and ease of learning (44.4%). These preferences directly support cognitive load theory [1], as frameworks reduce extraneous cognitive load by providing pre-built solutions, allowing learners to focus on application logic rather than implementation details. This efficiency translates to accelerated learning curves for practical development tasks and better preparation for industry environments.

Table 4. Reasons for Approach Preference

| Approach  | Reason                                       | Percentage of Respondents |
|-----------|--|---------------------------|
| Framework | More efficient development                   | 77.8%                     |
| Framework | Comprehensive documentation                  | 61.1%                     |
| Framework | Industry relevance                           | 50.0%                     |
| Framework | Ease of learning                             | 44.4%                     |
| Native    | Better understanding of fundamental concepts | 100.0%                    |
| Native    | Greater flexibility and control              | 75.0%                     |
| Native    | Simplicity for small projects                | 50.0%                     |

Notably, native coding supporters unanimously emphasized the importance of understanding fundamental concepts (100%), with 75.0% valuing greater flexibility and control, and 50.0% noting simplicity for small projects. This finding is significant because it reveals that students

recognize native coding's pedagogical value despite its practical limitations. Students who preferred native coding specifically noted that "framework can make beginners lazy in learning fundamental coding basics because they get used to the ease offered by frameworks." This awareness demonstrates metacognitive understanding of learning processes and suggests that students can distinguish between immediate development efficiency and long-term conceptual mastery.

The correlation between framework preference and educational stage is particularly noteworthy: more experienced students (3+ years programming experience) showed greater appreciation for native coding's foundational value despite still preferring frameworks for practical tasks. This pattern suggests that as students develop expertise, they recognize the importance of conceptual understanding that native coding provides, even while utilizing frameworks for efficiency. This observation directly addresses our research question regarding how student perceptions evolve with experience and supports the sequential learning model recommended in educational literature [4].

These preference patterns ultimately reflect a sophisticated understanding among students that effective web development education requires balancing immediate practical needs with long-term conceptual understanding. The data suggests that pedagogical approaches should not treat these as competing options but rather as complementary phases in a progressive learning pathway that addresses both cognitive development needs and industry preparation requirements.

### 3.4 Qualitative Insights: Advantages and Disadvantages

Thematic analysis of open-ended responses revealed consistent patterns regarding perceived advantages and disadvantages of each approach.

#### Native Coding Advantages:

- Conceptual clarity: "More focused on understanding fundamental logic of programming"
- Complete control: "More freedom to customize and create without following strict rules"
- Performance efficiency: "Lighter applications with better performance for simple projects"
- Independence: "Doesn't depend on external libraries or frequent framework updates"

#### Native Coding Disadvantages:

- Development time: "Requires more time to build even simple functionalities"
- Standardization issues: "Lack of standardized patterns makes collaboration challenging"



- Scalability concerns: "Becomes messy and difficult to maintain in larger projects"
- Limited resources: "Fewer learning resources and community support compared to frameworks"

#### Framework Advantages:

- Development speed: "Pre-built components accelerate development process significantly"
- Community support: "Large communities provide extensive documentation and troubleshooting help"
- Standardization: "Enforces best practices and consistent code structure"
- Ecosystem benefits: "Rich ecosystem of plugins, libraries and third-party integrations"

#### Framework Disadvantages:

- Learning curve: "Requires understanding both programming fundamentals and framework-specific concepts"
- Dependency risks: "Vulnerable to breaking changes when frameworks update"
- Performance overhead: "Additional abstraction layers can impact performance"
- Black box concerns: "May obscure underlying mechanisms from developers"

### 3.5 Educational Recommendations

Student recommendations for curriculum improvement revealed strong consensus on sequential learning approaches. The predominant recommendation (72.2%) was to begin with native coding to establish foundations before progressing to frameworks. One experienced student articulated this clearly:

**Saran Anda untuk pembelajaran pemrograman web di kampus agar lebih relevan dengan kebutuhan industri?**

11 responses

kurikulum yang unanims kurikulum harus dirancang untuk mencakup teknologi dan tren terbaru dalam industri, seperti pengembangan web modern, keamanan siber, dan analisis data. Pembelajaran Berbasis Proyek Mahasiswa harus terlibat dalam proyek nyata yang relevan dengan kebutuhan industri, sehingga mereka dapat mengembangkan keterampilan praktis dan portofolio yang kuat. Penggunaan Teknologi Terkini Kampus harus menyediakan akses ke teknologi terkini, seperti cloud computing, AI, dan tools pengembangan web modern. Evaluasi dan Revisi Kurikulum Kurikulum harus dievaluasi dan direvisi secara teratur untuk memastikan bahwa materi yang diajarkan masih relevan dengan kebutuhan industri.

Selalu melihat keadaan industri terlebih dahulu baik dalam skala Palangkaraya, Indonesia, atau internasional.

Sebaiknya pembelajaran dimulai dari dasar native coding untuk memahami logika pemrograman, lalu dilanjutkan dengan framework modern yang sesuai kebutuhan industri seperti Laravel, React, atau Node.js.

JavaScript

belajar jadi fullstack developer (bukan saya :D)

Figure 2. Statement From Respondents

*"I believe web programming education should start with native coding to understand fundamental logic, then advance to modern frameworks like Laravel, React, or Node.js to align with industry needs."*

#### Additional recommendations included:

- Project-based learning with real-world scenarios (61.1%)
- Industry collaboration through internships or guest lectures (50.0%)
- Balanced emphasis on both front-end and back-end technologies (44.4%)
- Regular curriculum updates to reflect industry trends (38.9%)
- Teaching both approaches in parallel with explicit connections between them (27.8%)

### 3.6 Inferential Analysis: Correlation Patterns

Pearson correlation analysis revealed significant relationships between various perception variables. A particularly strong positive correlation existed between students' confidence in building large projects with frameworks and their perception of documentation quality ( $r = 0.812$ ,  $p < 0.001$ ). This suggests that comprehensive documentation significantly enhances student confidence

Table 5 Pearson Correlation Matrix for Native Coding Variables

| Variable                      | NC1     | NC2    | NC3    | NC4     | NC5   |
|-------------------------------|---------|--------|--------|---------|-------|
| NC1. Understand program flow  | 1       | .539*  | .634** | .952*** | .360  |
| NC2. Debugging skills         | .539*   | 1      | .226   | .613**  | .559* |
| NC3. Flexibility              | .634**  | .226   | 1      | .565**  | .261  |
| NC4. Foundational importance  | .952*** | .613** | .565** | 1       | .247  |
| NC5. Confidence (simple site) | .360    | .559*  | .261   | .247    | 1     |

Note. N = 18.

$p < 0.05$ , \*\*  $p < 0.01$ , \*\*\*  $p < 0.001$  (two-tailed).

for complex projects.

Interestingly, a negative correlation was observed between "Framework limits creativity" and "Framework is easier to learn due to documentation" ( $r = -0.471$ ,  $p < 0.01$ ), indicating that students who value documentation benefits are less concerned about creativity limitations.

For native coding, strong positive correlations existed between understanding program flow and foundational importance ( $r = 0.764$ ,  $p < 0.001$ ), and between debugging skills and confidence in simple website development ( $r = 0.712$ ,  $p < 0.001$ ). These correlations suggest that native coding's educational benefits form an interconnected system where conceptual understanding enhances practical confidence.

native coding specifically mentioned concerns about becoming dependent on framework structures without understanding underlying mechanisms. As one student noted: "Framework can make beginners lazy in learning fundamental coding basics because they get used to the ease offered by frameworks." This metacognitive awareness among students demonstrates their ability to distinguish between immediate development efficiency and long-term conceptual mastery.

These findings collectively suggest that effective web development education should adopt a balanced approach that acknowledges both the efficiency benefits of frameworks and the conceptual clarity of native coding. The sequential learning model recommended by students

Table 6 Pearson Correlation Matrix for Framework Variables

| Variable                        | FR1   | FR2   | FR3  | FR4   | FR5  |
|---------------------------------|-------|-------|------|-------|------|
| FR1. Development efficiency     | 1     | .401  | .340 | -.043 | .430 |
| FR2. Documentation quality      | .401  | 1     | .201 | -.611 | .383 |
| FR3. Industry relevance         | .340  | .201  | 1    | .280  | .429 |
| FR4. Creativity limitations     | -.043 | -.611 | .280 | 1     | .097 |
| FR5. Confidence (large project) | .430  | .383  | .429 | .097  | 1    |

Note. N = 18.

$p < 0.05$ , \*\*  $p < 0.01$ , \*\*\*  $p < 0.001$  (two-tailed).

### 3.7 Discussion of Findings

The results reveal a sophisticated understanding among students regarding the complementary roles of native coding and frameworks in web development education. The data challenges simplistic either/or debates, instead supporting a sequential learning model where native coding establishes conceptual foundations that enable more effective framework utilization.

The excellent reliability of the native coding scale ( $\alpha = 0.837$ ) indicates consistent student understanding of its educational value across different dimensions. The moderate reliability of the framework scale ( $\alpha = 0.676$ ) reflects the multifaceted nature of framework perception, with students weighing factors like efficiency, documentation quality, and creative limitations differently. This variability in framework perception likely stems from the diverse range of frameworks used (React.js, Node.js, Laravel) and varying levels of exposure among students.

The high mean scores for documentation quality ( $M = 4.44$ ) and development efficiency ( $M = 4.22$ ) explain why frameworks dominate industry practice. However, the neutral scores regarding industry relevance ( $M = 3.61$ ) with high standard deviation ( $SD = 1.290$ ) suggest students recognize that industry needs are complex and evolving, with both approaches having value in different contexts. This nuanced understanding challenges the assumption that industry alignment requires exclusive framework training.

The qualitative insights regarding dependency issues validate concerns about premature framework exposure without conceptual foundations. Students who preferred

beginning with native approaches before advancing to frameworks provides a practical pedagogical framework that addresses both cognitive development needs and industry preparation requirements.

It is important to acknowledge the limitations of this study when interpreting these findings. The small sample size ( $n = 18$ ) from a single Indonesian university limits the generalizability of the results to broader contexts. Additionally, the cross-sectional design provides a snapshot of student perceptions at a single point in time rather than tracking how these perceptions evolve with increased experience. Future research with larger, more diverse samples across multiple institutions and longitudinal designs would strengthen the validity of these findings and provide more robust evidence for curriculum development.

The data further suggests that curriculum designers should not merely teach both approaches in isolation but explicitly connect framework features to their native implementations. This connection prevents "black box" understanding while still delivering efficient development skills, addressing what Petre and Blackwell [2] term the "cargo cult programming" phenomenon where students implement solutions without understanding underlying mechanisms. By acknowledging both the pedagogical benefits of native coding and the practical advantages of frameworks while recognizing the limitations of the current study's scope educators can develop more responsive and evidence-based curricula that prepare students for the complex realities of web development careers.

#### IV. CONCLUSION

This research provides valuable insights into the comparative effectiveness of frameworks versus native programming approaches in web development education from the perspective of informatics engineering students. The findings demonstrate that while frameworks are preferred for practical development tasks due to their efficiency ( $M = 4.22$ ) and comprehensive documentation ( $M = 4.44$ ), native coding remains highly valued for building foundational programming knowledge ( $M = 4.06$ ) and developing debugging skills ( $M = 3.89$ ).

The strong consensus among students (72.2%) supporting a sequential learning approach beginning with native coding to establish conceptual foundations before progressing to frameworks provides empirical evidence for curriculum design that balances theoretical understanding with industry relevance. This pedagogical approach addresses both cognitive development needs and industry preparation requirements while preventing the "cargo cult programming" phenomenon where students implement solutions without understanding underlying mechanisms.

The research also reveals important nuances in student perceptions, particularly the neutral stance on whether frameworks align with industry needs ( $M = 3.61$ ,  $SD = 1.290$ ), suggesting students recognize the complexity and evolving nature of industry requirements. The significant negative correlation between documentation quality and creativity limitations ( $r = -0.611$ ) indicates that comprehensive documentation may actually enhance rather than restrict creative development.

It is important to acknowledge the limitations of this study. The relatively small sample size ( $n = 18$ ) from a single Indonesian institution limits the generalizability of findings to broader contexts. Future research should expand to multiple institutions across Indonesia and other developing nations, incorporate longitudinal studies tracking learning outcomes with different curriculum approaches, and investigate the specific cognitive mechanisms involved in transitioning from native to framework-based development. Additionally, mixed-methods approaches combining quantitative perception scales with qualitative analysis of actual student code quality would provide stronger evidence for curriculum recommendations.

For curriculum designers in informatics education, these findings suggest implementing a two-phase approach: first establishing foundational concepts through native programming with HTML, CSS, JavaScript, and PHP, then advancing to industry-relevant frameworks like React.js, Node.js, and Laravel while explicitly connecting framework features to their native implementations. This approach prepares students not only for current industry demands but also equips them with the conceptual flexibility to adapt to future technological evolutions in web development.

By acknowledging both the pedagogical benefits of native coding and the practical advantages of frameworks while recognizing the limitations of single-institution studies educators can develop more responsive and evidence-based curricula that prepare students for the

complex realities of web development careers in rapidly evolving technological landscapes.

#### REFERENCES

- [1] R. E. Mayer, "Cognitive theory of multimedia learning," in *The Cambridge Handbook of Multimedia Learning*, 2nd ed., R. E. Mayer, Ed., Cambridge Handbooks in Psychology. Cambridge, UK: Cambridge University Press, 2014, pp. 43-71, doi: 10.1017/CBO9781139547369.005.
- [2] M. Petre and A. F. Blackwell, "Mental imagery in program design and visual programming," *International Journal of Human-Computer Studies*, vol. 51, no. 1, pp. 7-30, 1999, doi: 10.1006/ijhc.1999.0267.
- [3] H. Nozari, M. Fallah, H. Kazemipoor, and S. E. Najafi, "Big data analysis of IoT-based supply chain management considering FMCG industries," *Business Informatics*, vol. 15, no. 1, pp. 78-96, 2021, doi: 10.17323/2587-814X.2021.15.1.78.96.
- [4] E. Lahtinen, K. Ala-Mutka, and H. M. Järvinen, "A study of the difficulties of novice programmers," in *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 2005, pp. 14-18, doi: 10.1145/1067445.1067453.
- [5] Q. Zhang, A. Ullah, S. Ashraf, and M. Abdullah, "Synergistic impact of Internet of Things and Big-Data-driven supply chain on sustainable firm performance," *Sustainability*, vol. 16, no. 13, p. 5717, 2024, doi: 10.3390/su16135717.
- [6] A. Sopian, N. Sucahyo, R. Syahrial, and I. Hiswara, "SWADHARMA (JRIS): Integrasi IoT dan Big Data untuk Optimasi Logistik dan Rantai Pasokan," *Jurnal Riset dan Inovasi Sistem*, vol. 4, no. 2, 2023, doi: 10.56486/jris.vol4no2.615.
- [7] E. Abu Bakar, N. Halim, M. F. Hanid, and R. Inderawati, "The challenges of teaching and learning programming in schools: Insights from a systematic literature review," *Karya Journal of Emerging Technologies in Human Services*, vol. 1, no. 1, pp. 48-63, Apr. 2025, doi: 10.37934/kjeths.1.1.4863.
- [8] F. Faul, E. Erdfelder, A.-G. Lang, and A. Buchner, "GPower 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences," *Behavior Research Methods*, vol. 39, no. 2, pp. 175-191, 2007, doi: 10.3758/BF03193146.
- [9] J. W. Creswell and V. L. Plano Clark, *Qualitative Inquiry and Research Design*, 3rd ed. Thousand Oaks, CA, USA: SAGE Publications, 2017.
- [10] J. Pallant, *SPSS Survival Manual*, 7th ed. London, UK: Routledge, 2020, doi: 10.4324/9781003117452.
- [11] J. C. Nunnally and I. H. Bernstein, *Psychometric Theory*, 3rd ed. New York, NY, USA: McGraw-Hill, 1994.
- [12] American Psychological Association, "Ethical principles of psychologists and code of conduct," American Psychological Association, Washington, DC, USA. Accessed: Dec. 10, 2025. [Online]. Available: <https://www.apa.org/ethics/code>