

PAPER

BookEZ Case-to-System Learning Model for Integrated SAD and Web Development

Ainun Rafieza Ahmad
Tajuddin*, Nur Elyani
Mohammad, Nur Nadia
Musa

Certificate in Information
Technology Programme, Kolej
Komuniti Arau, Arau, Perlis,
Malaysia

Correspondence Author Email:
ainunrafieza@staf.kkarau.edu.my

ABSTRACT

System Analysis and Design (SAD) helps students structure a real problem before a system is developed. However, classroom observation among certificate-level information technology students indicates that many learners can understand a case-study narrative but struggle to translate the problem into coherent SAD documentation. The BookEZ Case-to-System Learning Model was developed as a complete worked learning reference that connects an authentic room-booking case study, syllabus-aligned SAD artefacts and a functional web-based system. Its key feature is a progressive side-by-side learning approach: while students construct or examine each SAD artefact, they simultaneously explore the related BookEZ screen or workflow to visualise how the planning decision appears in an actual system. The model was developed through curriculum mapping, case construction, documentation preparation, web-system implementation and guided reflective mapping. A rubric-based preliminary comparison across two cohorts ($n = 32$) showed improvement in logical flow, diagram completeness, symbol accuracy, ERD-database alignment and documentation organisation. The overall mean project score increased from 66% to 82%. The model also supports learning continuity when students progress to Web Development and need to plan a project before coding. This paper presents the model, its curriculum alignment, implementation workflow, preliminary evidence and directions for further validation.

KEYWORDS

Case-to-System Learning; Progressive Side-by-Side Learning; System Analysis and Design; Web Development; TVET; Worked Example

How to Cite: [1] A. R. A. Tajuddin, N. E. Mohammad, and N. N. Musa, "BookEZ Case-to-System Learning Model for Integrated SAD and Web Development," *Innovation and Creativity Product Exhibition*. vol. 1, no. 1, pp. 43-50, 2026.

Article Submitted 2026-06-01. Revision Uploaded 2026-06-12. Final Acceptance 2026-06-30

© by the authors of this article. Published under CC-BY.

1. INTRODUCTION

Technical and vocational education and training (TVET) is expected to develop knowledge that can be applied in authentic working contexts. UNESCO's TVET strategy for 2022–2029 emphasises skills development for productive employment and transitions towards increasingly digital economies [2]. The OECD similarly highlights the central role of vocational pathways in education systems [3], while the World Economic Forum identifies technological literacy, analytical thinking and continuous learning as increasingly important workplace capabilities [4]. For information technology students, these expectations require more than an ability to reproduce technical procedures. Students must be able to understand a problem, identify relevant requirements, organise a solution and connect their planning decisions to a functioning digital product.

System Analysis and Design (SAD) provides a structured foundation for this process. Requirements engineering standards describe the importance of establishing requirements-related processes and information items throughout the system and software life cycle [5]. Systems analysis literature likewise positions preliminary investigation, requirements identification, modelling, user-interface planning, testing and implementation as connected activities rather than independent tasks [6]–[8]. In the Certificate in Information Technology programme at Kolej Komuniti Arau, the SFC11643 System Analysis and Design Fundamentals syllabus requires students to explain appropriate SAD concepts and construct key deliverables for the phases of the System Development Life Cycle (SDLC) using suitable tools and methods [1]. Its content includes preliminary investigation, system requirements, fact-finding, Data Flow Diagrams (DFD), data dictionaries, process descriptions, user-interface design, input and output design, validation rules, testing, documentation and system implementation [1].

Despite this structured syllabus, an instructional difficulty emerged during classroom practice. Students were generally able to read a case study and understand the broad operational problem described in the scenario. The difficulty arose when they were asked to convert that understanding into formal SAD documentation. For novice learners, the transition from an everyday explanation of a problem to a structured system representation is not automatic. They may recognise that users face delays or confusion, but still struggle to define system scope, identify actors and requirements,

distinguish input from process and output, organise a DFD or explain how a planned interface responds to the case-study problem. When each artefact is taught separately, some students treat the activity as a diagramming exercise instead of viewing it as one step in a coherent system-development process.

This issue becomes more visible when students progress to Web Development in the following semester. Without a clear understanding of why planning precedes implementation, students may begin coding immediately and add functions incrementally without a stable view of the system's purpose, workflow and user needs. The result is not necessarily a lack of effort. Rather, the development process becomes less organised because the conceptual bridge between problem analysis and implementation has not yet become visible to the learner. The teaching challenge is therefore broader than a simple diagram-to-code mismatch. The more fundamental gap is the student's ability to translate a real case into SAD documentation, observe how each artefact is reflected in a functioning system and later apply the same planning logic to a new web project.

Project-based learning (PjBL), case-based learning and example-based learning provide a strong pedagogical basis for responding to this challenge. A meta-analysis by Zhang and Ma found that PjBL makes a moderate positive contribution to learning effectiveness, particularly when authentic problems, planning and project processes are clearly structured [9]. Wijnia et al. reported positive motivational effects for student-centred problem-driven approaches, including project-based and case-based learning [10]. Zhang, Guan and Hu further found that PjBL can support computational-thinking dimensions such as critical thinking, algorithmic thinking and problem solving [11]. In software-engineering education, Morais, Ferreira and Veloso demonstrated the value of project-based learning across connected curricular units [12], while Zhang, Zhang and Zhang showed that case-based teaching helps learners connect theory with its application in computer-science courses [13]. Worked examples and scaffolding are especially relevant for novice learners. Scaffolding regulates the amount and sequence of information presented to learners so that they can focus on meaningful relationships instead of becoming overwhelmed by multiple unfamiliar tasks [18], [19]. In software-engineering education, example-based learning has also been identified as a promising approach for supporting interpretation, application, confidence and student interaction [16]. Nevertheless, a gap remains in the local teaching context: students need one complete, syllabus-aligned reference that begins with an understandable case and allows them to construct each SAD artefact while viewing the related part of a completed web system. A concurrent visual and functional reference can make abstract modelling decisions more concrete without reducing the activity to imitation.

The BookEZ Case-to-System Learning Model was developed to address this gap. BookEZ is a web-based room-booking system used as an authentic worked reference. The innovation is not limited to the web application itself. It is a structured teaching package consisting of a realistic case study, aligned SAD artefacts, a completed BookEZ system and guided prompts that connect each artefact to a visible part of the implementation. Its distinctive feature is progressive side-by-side learning. During a SAD lesson, students can work on a context diagram, DFD, data dictionary or interface-planning activity while simultaneously opening the relevant BookEZ screens and workflows. The system acts as a visual and functional scaffold that helps learners imagine the system boundary, external entities, data exchanges, processes, input fields and outputs more clearly. During Web Development, the same example reinforces the importance of planning before coding. This article presents the development, curriculum alignment, implementation pathway, educational value and validation plan of the BookEZ Case-to-System Learning Model.

2. METHOD

2.1 Innovation Context and Development Approach

The BookEZ Case-to-System Learning Model was developed as a pedagogical innovation for certificate-level information technology students. The development process used a curriculum-aligned worked-example approach supported by project-based, case-based and scaffolded-learning principles. Rather than presenting SAD artefacts as independent classroom tasks, the innovation was organised as a complete learning sequence. The sequence begins with an authentic operational problem and then applies a progressive side-by-side approach: students construct or examine a syllabus-aligned SAD artefact while simultaneously exploring the corresponding BookEZ screen or workflow. Guided questioning helps learners explain how the visible system relates to the underlying analysis and planning decision. The learning sequence later supports the planning of independent web-development projects.

The development method consisted of six stages. First, the SFC11643 syllabus was analysed to identify the concepts and deliverables that students must understand. Second, a realistic room-booking case study was structured to include operational issues, user needs and system boundaries. Third, SAD reference documents were prepared from the case study. Fourth, a functional BookEZ web system was developed from the same requirements and planned workflows. Fifth, a progressive side-by-side mapping pathway was created so that each SAD topic could be learned together with a related system screen, feature or workflow. Sixth, guided reflection prompts and an evidence-collection plan were prepared for classroom implementation and future validation.

2.2 Curriculum Mapping

Table 1 shows the direct alignment between the official SAD syllabus and the worked-reference components included in BookEZ. The mapping prevents the innovation from becoming a generic web-development demonstration. Each component is included because it supports a specific teaching need in the SAD course.

Table 1. Curriculum alignment of the BookEZ worked reference

SFC11643 syllabus area	BookEZ worked-reference component	Learning purpose
Preliminary investigation	Room-booking operational case and problem summary	Help students define the problem, opportunity, scope and constraints.
System requirements	Structured input, output, process, performance and control requirements	Show how an everyday problem becomes a formal requirement set.
Fact-finding	Suggested interview questions, observation prompts and information summary	Demonstrate how evidence supports analysis decisions.
Data and process modelling	Context diagram, Level 0 DFD, data dictionary and process descriptions	Help students construct data-flow artefacts while exploring related BookEZ screens and workflows to visualise entities, data movement and process logic.
User-interface design	Reference screens and interface rationale	Relate user needs to visible screen elements and controls through side-by-side comparison.
Input and output design	Form fields, validation rules and sample outputs	Explain why fields, controls and outputs are planned by comparing the document with the implemented form and display.
Testing and documentation	Test-case examples and user guidance	Connect development decisions to quality assurance and user support.

2.3 Progressive Side-by-Side Case-to-System Workflow

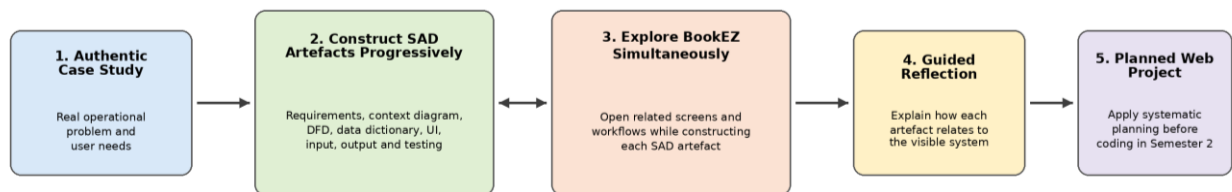


Figure 1. BookEZ progressive side-by-side Case-to-System learning model

As shown in Figure 1, the model begins with an authentic case study and then places SAD documentation and the completed BookEZ system in an active two-way relationship. Students do not need to wait until every document has been completed before viewing the website. Instead, they construct or examine each SAD artefact progressively while exploring the related BookEZ screen or workflow. For example, while preparing a context diagram, students can inspect the website to identify user roles, major interactions and the information exchanged with the system. Guided reflection then helps students explain why each visible function exists and how it is represented in the documentation. Finally, the same planning logic is transferred to a new Web Development project in the following semester.

2.4 Innovation Components and Tools

The innovation package is modular so that the lecturer can use selected materials according to the weekly topic while still retaining a coherent overall case. Table 2 summarises the main components.

Table 2. Components of the BookEZ innovation package

Component	Description	Teaching use
Authentic case study	A realistic room-booking scenario with identifiable operational problems and user needs.	Used as the common starting point for analysis activities.
SAD documentation pack	Syllabus-aligned examples of requirements, DFD, data dictionary, process descriptions, UI planning, input/output planning and test considerations.	Used as worked examples, discussion materials and comparison references.
Completed BookEZ web system	A functional web-based system developed from the same room-booking scenario.	Used as a concurrent visual and functional scaffold while students construct or examine each SAD artefact.
Progressive side-by-side mapping guide	Topic-based prompts that pair each SAD artefact with a related BookEZ screen, feature or workflow.	Used during class so that students can visualise the meaning of the artefact while constructing or critiquing it.

Component	Description	Teaching use
Guided reflection and cross-semester checklist	Questions for reverse mapping and a planning checklist for later Web Development projects.	Used to explain why a feature exists and to encourage systematic planning before coding.
Digital delivery platform	A central online location for the case, documentation, BookEZ system access and guided activities.	Used for repeated side-by-side access during class and later revision.

2.5 Validation Method and Evidence Boundary

The initial implementation was evaluated through a rubric-based comparison involving two comparable cohorts of certificate-level information technology students ($n = 32$). The rubric examined downstream indicators of structured SAD work and implementation coherence, namely logical flow and organisation in DFD, diagram completeness, correct use of modelling symbols, ERD-database consistency and documentation organisation. Overall mean project scores were also compared. These indicators were retained because they were derived from actual student assessment records and provide preliminary evidence that a structured worked reference can improve the coherence of student outputs.

The existing evidence should nevertheless be interpreted within its scope. It demonstrates improvement in the quality and consistency of student artefacts, but it does not yet measure every dimension of the refined progressive side-by-side learning approach directly. Future implementation cycles should add rubric items for the accuracy of problem translation, requirement completeness, context-diagram clarity and project-planning quality, together with short learner reflections and lecturer reflective notes. These additional instruments can show more precisely whether simultaneous exploration of the completed BookEZ system helps students visualise the meaning of each SAD artefact.

3. RESULT AND DISCUSSION

3.1 Developed Innovation Output

The principal result of this work is a reusable pedagogical model that reorganises the teaching of SAD around one complete worked reference. The model does not replace the existing syllabus. Instead, it provides continuity across syllabus topics by showing that the same operational problem can be examined repeatedly from different analytical perspectives. Its distinctive contribution is that the completed BookEZ website is not reserved only as an endpoint. It is also used as a visual and functional scaffold while students construct individual SAD artefacts. By moving repeatedly between the case, the documentation and the related website feature, each new artefact becomes another view of the same system rather than a separate academic task.

The first output is the authentic BookEZ case study. The case describes a room-booking situation in which manual or poorly structured processes can result in uncertainty, duplicate requests, delays and difficulty monitoring booking information. The scenario is deliberately understandable to certificate-level learners. It allows the lecturer to begin with an everyday operational problem before introducing technical terminology. Students can discuss who uses the process, what information is required, what decisions are made and what outputs are expected. This starting point responds directly to the classroom observation that students often understand the story but require support to convert the story into a formal system view.

The second output is the SAD documentation pack. The documentation is aligned with the SFC11643 syllabus and focuses on the artefacts that students are expected to construct or explain. The pack includes a preliminary problem summary, scope and constraints, structured system requirements, fact-finding prompts, a context diagram, Level 0 DFD, data dictionary, process descriptions, interface-planning references, input and output considerations, validation examples, test-case examples and documentation notes. This alignment is significant because the earlier version of the innovation narrative placed excessive emphasis on database mapping and coding. Those elements remain useful for the Web Development transition, but the central SAD teaching problem occurs earlier: learners need to understand how a real case becomes structured documentation.

The third output is the completed BookEZ web system. Its purpose in the innovation is pedagogical. It functions both as a visible implementation and as a concurrent reference during SAD activities. When students construct a context diagram, they can explore the website to identify user roles, major interactions and the information exchanged with the system. When a DFD is discussed, students can observe how user actions relate to processes and outputs. When an input form is analysed, the completed website makes fields, controls and validations visible. When interface design is introduced, students can compare screen elements with the needs identified in the case study. The system therefore makes otherwise abstract documentation easier to imagine and interpret.

The fourth output is a set of progressive side-by-side mapping and guided reflection activities. Students are not merely shown a finished website and they are not expected to copy it mechanically. Instead, the lecturer selects the relevant BookEZ screen or workflow while students construct, examine or critique a SAD artefact. Students are prompted to ask why each feature exists, which user requirement it addresses, where the related process appears in the DFD, what input data must be captured and which output is generated. This two-way movement between documentation and implementation turns the completed system into an analytical teaching aid.

3.2 A Curriculum Bridge Across Two Semesters

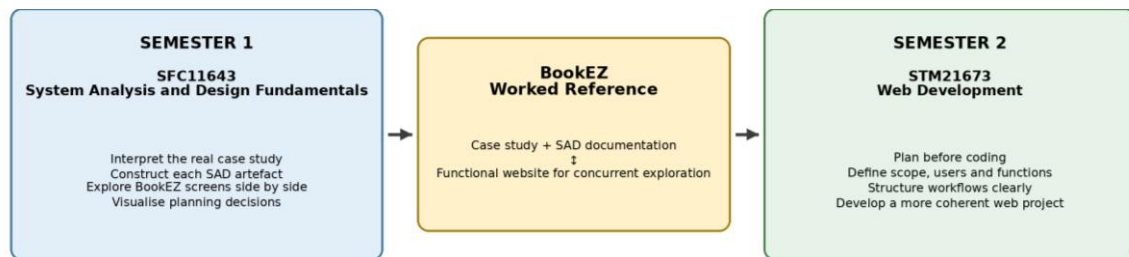


Figure 2. Cross-semester learning continuity supported by BookEZ

Figure 2 illustrates the model's cross-semester role. During Semester 1, BookEZ supports the teaching of SFC11643 System Analysis and Design Fundamentals. The emphasis is on translating a case into documentation while simultaneously exploring related parts of the completed system. This side-by-side process helps students visualise how SAD deliverables work together and why each artefact matters. During Semester 2, the same reference can be revisited in STM21673 Web Development. At this stage, the earlier planning logic reinforces the message that development should be organised before coding starts. The model is therefore not a claim that the two courses are merged into a single course. Instead, it provides pedagogical continuity between two sequential learning experiences.

This distinction strengthens the innovation. The product is not simply a room-booking website and it is not merely a set of diagrams. Its value lies in the relationship among the case study, the SAD documentation, the completed implementation and the subsequent student project-planning practice. Integrated project-based curricula have previously been used to create stronger continuity across separate software-engineering courses [12], [17]. Similarly, real-demand project approaches show that requirements gathering, development, testing, deployment and maintenance can be organised as connected steps [14]. BookEZ adapts this logic for certificate-level learners using a controlled, lecturer-prepared example that reduces complexity while preserving the relationship between planning and implementation.

The controlled example is particularly appropriate for novice learners. A completely open-ended project can be motivating, but it can also require students to make too many decisions simultaneously. Cognitive load theory explains that novices benefit when instructional support helps them focus on relevant relationships [18], [19]. The BookEZ model provides this support by giving students a stable case and a visible system reference that can be consulted while each artefact is learned. The intention is not to encourage copying. Instead, the website helps students imagine the meaning of an abstract document before the support is gradually faded as they become more capable of planning their own systems.

3.3 Learning Value of the Case-to-System Sequence

The learning value of the model can be discussed through four connected transitions. Table 3 summarises the difficulty addressed at each transition and the support provided by BookEZ.

Table 3. Learning transitions supported by the BookEZ model

Learning transition	Common learner difficulty	BookEZ support	Expected learning evidence
Case study → system problem	Students describe the story but do not define the system boundary clearly.	Problem summary, scope prompts and user-needs discussion.	A clearer problem statement and justified scope.
System problem → SAD artefact with concurrent website reference	Students struggle to identify requirements and organise models coherently when the representation remains abstract.	Worked SAD examples paired with related BookEZ screens and workflows.	Artefacts that are logically connected to the case and explained using visible system evidence.
SAD artefact ↔ completed system	Students may treat documentation as academic paperwork with no visible purpose.	Progressive side-by-side exploration and guided reverse mapping.	Ability to explain how each artefact relates to an implemented feature or workflow.
Completed reference → own web project	Students may begin coding before planning the system.	Cross-semester checklist and reverse-mapping questions.	A more structured project plan before coding begins.

The first transition is from the case-study narrative to a system problem. This stage is essential because the student's initial understanding is often expressed in conversational terms. The lecturer uses the BookEZ scenario to guide students towards a more precise problem statement, scope and set of constraints. This process is consistent with the official syllabus requirement to understand the problem or opportunity and define project scope and constraints during preliminary investigation [1]. It also reflects requirements-engineering practice, where business or mission needs must be clarified before detailed system requirements are produced [5].

The second transition is from the system problem to SAD documents with a simultaneous system reference. The same case is used to identify input, output, process, performance and control requirements. Students then examine the

movement of information through a context diagram and Level 0 DFD. While constructing these artefacts, they can open the related BookEZ pages and workflows to clarify who interacts with the system, what information enters or leaves the system and how a process produces an output. The data dictionary and process descriptions are not introduced as unrelated paperwork; they clarify the meaning of data flows and processes already discussed. This sequencing is aligned with systems-analysis texts that emphasise the connected role of investigation, modelling and solution planning [6]–[8].

The third transition is an iterative relationship between each SAD artefact and the completed BookEZ system. The implementation adds value to SAD instruction because it gives students something visible and functional to examine while they are still learning the documentation. A planned input can be compared with an actual form field, a validation rule can be observed during user interaction, an output requirement can be related to a displayed result and a DFD process can be connected to a workflow. A context diagram becomes easier to imagine when students can inspect the types of users and major system interactions. The implementation therefore gives meaning to the document while the document explains the logic behind the implementation. Case-based learning research supports this approach because cases can help students understand how theoretical knowledge is applied and can promote analysis and reflection [13].

The fourth transition is from a completed reference to the learner's own web project. When students enter Web Development, BookEZ can be reused as a planning reference. The lecturer can ask students to define their project's problem, users, functions, inputs, processes and outputs before coding. This reduces the risk of building pages without a coherent workflow. Example-based learning research in software-engineering education similarly suggests that examples can support learners in interpreting and applying concepts while increasing confidence [16].

3.4 Preliminary Implementation Results and Evidence Boundary

Preliminary implementation evidence was obtained from a rubric-based comparison across two comparable cohorts ($n = 32$). Table 4 summarises the recorded results. The overall mean project score increased from 66% before implementation to 82% after implementation. Improvements were also observed across all five rubric components. The strongest increases were recorded for ERD-database alignment, which increased by 38 percentage points, and logical flow, which increased by 34 percentage points.

The results support the value of using one coherent case, aligned SAD documentation and a completed web-system reference as an integrated learning pathway. Improvements in logical flow, diagram completeness, symbol accuracy and documentation organisation indicate that students were better able to produce structured artefacts. The improvement in ERD-database alignment is treated as a supplementary downstream indicator: it shows that the planning process can support the later transition from SAD documentation to implementation, even though database alignment is not the sole focus of the refined innovation.

The preliminary evidence must still be interpreted cautiously. The recorded rubric data demonstrate improvement in student outputs, but the refined progressive side-by-side feature should be validated more directly in future classroom cycles. Additional evidence should examine whether students identify system boundaries, external entities, data flows and requirements more accurately while exploring the relevant BookEZ screen or workflow. Learner reflections and lecturer reflective notes can also determine whether the website reduces repeated explanation and helps students visualise abstract SAD concepts more clearly.

Table 4. Preliminary rubric-based implementation results

Rubric component	Before (%)	After (%)	Change (percentage points)
Logical flow and organisation (DFD)	54	88	+34
Diagram completeness	58	84	+26
Correct use of modelling symbols	62	91	+29
ERD-database alignment	49	87	+38
Documentation organisation	56	83	+27
Overall mean project score	66	82	+16

The table is based on actual assessment evidence from the earlier implementation. Because the innovation has now been described more precisely as a progressive side-by-side learning model, subsequent validation should retain the existing rubric indicators and add direct measures of case-to-document translation and learner visualisation.

3.6 Progressive Side-by-Side Classroom Facilitation

The model is intended to support active teaching rather than passive demonstration. A completed web system alone does not guarantee understanding. The lecturer uses the website strategically while students construct or examine each SAD artefact. Table 5 presents a progressive side-by-side facilitation sequence that can be adapted to the weekly lesson plan.

Table 5. Suggested progressive side-by-side classroom facilitation sequence

Teaching phase	Lecturer action	Student learning activity	Transfer focus
Problem discovery	Present the room-booking case and allow a short initial interpretation before using the website as a reference.	Identify operational issues, users, information needs and possible constraints.	Move from everyday language to a defined system problem.

Teaching phase	Lecturer action	Student learning activity	Transfer focus
Artefact construction	Introduce the relevant SAD artefact and open the related BookEZ screen or workflow alongside the activity.	Construct or critique requirements, context-diagram elements, DFD processes, dictionary entries or interface decisions while exploring the system.	Visualise an abstract SAD concept through a concrete functional reference.
Side-by-side questioning	Ask students to explain how the visible feature relates to the artefact being learned.	Identify entities, inputs, outputs, processes, controls and validation rules using evidence from the website.	Connect planning decisions to visible implementation.
Reverse mapping and reflection	Ask students to trace a selected screen or workflow back to the supporting documentation.	Explain why the feature exists and which requirement or process supports it.	Strengthen reflective understanding of SAD.
Independent transfer	Gradually reduce the reference support and provide a new case or student web-project idea.	Prepare a basic plan before coding begins.	Apply the Case-to-System thinking process independently.

The first stage remains important because students should attempt to understand the operational problem before treating the completed website as an answer key. Once the initial problem has been discussed, the lecturer opens the relevant BookEZ page or workflow alongside the SAD activity. For example, while constructing a context diagram, students can inspect the website to identify external entities, major interactions and the data entering or leaving the system. While constructing a Level 0 DFD, they can trace visible actions such as submitting a booking request or checking a booking status and discuss how these actions relate to processes, data stores and outputs.

The side-by-side reference should be accompanied by questioning. Students can be asked why a field is required, what output is generated, which validation rule reduces error or how a visible workflow relates to a DFD process. The support should then be reduced gradually. At the beginning, students can use complete BookEZ examples. Later, selected documentation elements or mapping prompts can be omitted so that they make more decisions independently. This maintains the analytical value of the activity while preparing students to plan a different web project before coding.

3.7 Novelty, Replication and Commercial Potential

The novelty of BookEZ lies in its curriculum-aligned Case-to-System sequence and progressive side-by-side learning approach. The innovation begins at the point where the learner experiences difficulty: converting an understood case-study problem into structured SAD documentation. Unlike an approach that displays the final website only after all documentation is completed, BookEZ allows students to examine the related screen or workflow while constructing each SAD artefact. The website operates as a visual and functional scaffold, while guided questioning prevents passive copying. The model also differs from a generic project-based activity because the worked reference is deliberately mapped to the official SAD syllabus and is reused to support systematic planning in the following Web Development course.

The model has practical replication potential. A lecturer can adapt the same structure to another domain such as inventory management, appointment booking, student services or community facilities. The reusable format would remain consistent: prepare an authentic case, produce syllabus-aligned SAD artefacts, develop a completed reference system, create a side-by-side mapping guide, design reflection prompts and use a project checklist for the subsequent development course. In this form, the innovation can be distributed as a modular digital teaching kit.

Commercialisation should be positioned realistically. The immediate product is a lecturer-ready educational package rather than a large-scale software-as-a-service platform. A feasible package could include the BookEZ case study, editable document templates, completed reference documentation, web-system demonstration access, side-by-side mapping worksheets, reflection prompts, a lecturer implementation guide and assessment-support materials. A future training workshop could guide lecturers in constructing comparable Case-to-System learning references for their own contexts. Wider institutional adoption should follow pilot validation, expert review and refinement based on actual learner evidence.

The model also has limitations. The current BookEZ system represents one domain and may not expose every type of system requirement. A completed reference can become overly prescriptive if students are encouraged to imitate rather than analyse. The lecturer must therefore use questioning, comparison and gradually reduced support so that the website remains a scaffold rather than an answer key. Moreover, the cross-semester impact must be evaluated over time. Evidence should examine not only whether students complete SAD documents more accurately, but also whether they apply the planning approach independently when developing a new web project.

4. CONCLUSION

The BookEZ Case-to-System Learning Model was developed to address a specific instructional difficulty among certificate-level information technology students: learners may understand the problem described in a real case study but still struggle to translate that understanding into coherent System Analysis and Design documentation. The innovation

organises one authentic room-booking case, syllabus-aligned SAD artefacts, a completed web-based BookEZ system and guided activities into a reusable teaching pathway. Its distinctive contribution is progressive side-by-side learning. While students construct or examine an artefact such as a context diagram, DFD, data dictionary or interface plan, they can simultaneously explore the related BookEZ screen or workflow. This helps students visualise system boundaries, external entities, data exchanges, processes, inputs, outputs and interface elements more concretely. The website is therefore not treated merely as an endpoint or a product showcase; it functions as a visual and functional scaffold, while guided questioning ensures that students continue to analyse rather than copy. Preliminary rubric-based comparison across two cohorts ($n = 32$) showed improvement across logical flow, diagram completeness, symbol accuracy, ERD-database alignment and documentation organisation, with the overall mean project score increasing from 66% to 82%. These results provide encouraging evidence of improved coherence in student outputs. Future validation should add direct measures of problem translation, context-diagram clarity, learner reflection and independent project planning during Web Development. By connecting an understandable case to documentation and a visible working system during the same learning activity, BookEZ offers a practical way to make SAD learning more concrete, coherent and transferable across semesters.

REFERENCES

- [1] Department of Polytechnic and Community College Education, "SFC11643 System Analysis and Design Fundamentals," Course Information, Version 04102023_3, effective Session I 2025/2026.
- [2] UNESCO, Transforming Technical and Vocational Education and Training for Successful and Just Transitions: UNESCO Strategy 2022–2029. Paris, France: UNESCO, 2022.
- [3] OECD, Education at a Glance 2023: OECD Indicators. Paris, France: OECD Publishing, 2023.
- [4] World Economic Forum, The Future of Jobs Report 2025. Geneva, Switzerland: World Economic Forum, 2025.
- [5] ISO/IEC/IEEE, ISO/IEC/IEEE 29148:2018 Systems and Software Engineering—Life Cycle Processes—Requirements Engineering. Geneva, Switzerland: ISO, 2018.
- [6] A. Dennis, B. H. Wixom and R. M. Roth, Systems Analysis and Design, 8th ed. Hoboken, NJ, USA: Wiley, 2021.
- [7] J. W. Satzinger, R. B. Jackson and S. D. Burd, Systems Analysis and Design in a Changing World, 8th ed. Boston, MA, USA: Cengage Learning, 2020.
- [8] J. A. Hoffer, J. F. George and J. S. Valacich, System Analysis and Design, 12th ed. Harlow, UK: Pearson, 2023.
- [9] L. Zhang and Y. Ma, "A study of the impact of project-based learning on student learning effects: A meta-analysis study," *Frontiers in Psychology*, vol. 14, Art. no. 1202728, 2023, doi: 10.3389/fpsyg.2023.1202728.
- [10] L. Wijnia, G. Noordzij, L. R. Arends, S. M. M. Loyens and R. M. J. P. Rikers, "The effects of problem-based, project-based, and case-based learning on students' motivation: A meta-analysis," *Educational Psychology Review*, vol. 36, Art. no. 29, 2024, doi: 10.1007/s10648-024-09864-3.
- [11] W. Zhang, Y. Guan and Z. Hu, "The efficacy of project-based learning in enhancing computational thinking among students: A meta-analysis of 31 experiments and quasi-experiments," *Education and Information Technologies*, vol. 29, pp. 14513–14545, 2024, doi: 10.1007/s10639-023-12392-2.
- [12] P. Morais, M. J. Ferreira and B. Veloso, "Improving student engagement with project-based learning: A case study in software engineering," *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje*, vol. 16, no. 1, pp. 21–28, 2021, doi: 10.1109/RITA.2021.3052677.
- [13] X. Zhang, B. Zhang and F. Zhang, "Student-centered case-based teaching and online–offline case discussion in postgraduate courses of computer science," *International Journal of Educational Technology in Higher Education*, vol. 20, Art. no. 6, 2023, doi: 10.1186/s41239-022-00374-2.
- [14] L. H. Silva, R. X. Castro and M. C. Guimaraes, "Supporting real demands in software engineering with a four steps project-based learning approach," *arXiv:2102.01631*, 2021.
- [15] B. R. N. Oliveira and E. Y. Nakagawa, "Towards a case-based learning approach to support software architecture education," *arXiv:2210.04794*, 2022.
- [16] T. P. Bonetti, W. Silva and T. E. Colanzi, "Example-based learning in software engineering education: A systematic mapping study," *arXiv:2503.18080*, 2025.
- [17] Y. Afshar, S. El Ariss and J. May, "An integrated software engineering curriculum through project-based learning (PBL)," in *Proc. ASEE Annual Conference & Exposition*, 2022.
- [18] C. C. A. van Nooljen, T. van Gog, A. M. M. Muijselaar and F. Paas, "A cognitive load theory approach to understanding expert scaffolding of visual problem-solving tasks: A scoping review," *Educational Psychology Review*, vol. 36, 2024, doi: 10.1007/s10648-024-09848-3.
- [19] J. Sweller, J. J. G. van Merriënboer and F. Paas, "Cognitive architecture and instructional design: 20 years later," *Educational Psychology Review*, vol. 31, pp. 261–292, 2019, doi: 10.1007/s10648-019-09465-5.
- [20] J. J. G. van Merriënboer and P. A. Kirschner, *Ten Steps to Complex Learning: A Systematic Approach to Four-Component Instructional Design*, 3rd ed. New York, NY, USA: Routledge, 2018.
- [21] J. Biggs, C. Tang and G. Kennedy, *Teaching for Quality Learning at University*, 5th ed. Maidenhead, UK: Open University Press, 2022.
- [22] X. Franch, M. Glinz, D. Mendez and N. Seyff, "A study about the knowledge and use of requirements engineering standards in industry," *arXiv:2105.13961*, 2021.
- [23] R. Garcia, C. Treude and A. Valentine, "Application of collaborative learning paradigms within software engineering education: A systematic mapping study," *arXiv:2310.18845*, 2023.