Phase	Process	Description
Analysis phase	Validate the performance gap	Defining the reasons for the performance gap and generating a purpose statement. The difference between the actual and the required performance should be measured, and then possible reasons should be explored, which can be a lack of resources, motivation, knowledge, or skill. If it is a lack of knowledge or skill, then a clear purpose statement should be formulated to increase the learner's performance to reach the required performance.
	Determine Instructional Goals	Deriving clear goal statements that cover the performance gap. Instructional goals should show what the learners can do after participating in the course. Bloom's taxonomy is one of the widely used approaches to organize the cognitive domains of the learning process and define effective goals.
	Confirm the Intended Audience	Analyzing the course learners, which will significantly impact decision-making in the following phases. The audience's experiences, skills, location, number, and attitudes are the most required information.
	Identify Required Resources	Defining all necessary resources to complete the ADDIE model. Different types of resources have to be clarified. Reusing the previous learning material and strategies is a part of the content resources. It is essential to check the number of computers or flip charts to define the available technology resources. Instructional facilities that significantly impact the following phases, such as the number of available rooms, the number of expected students, and the tools used in digital learning, must be defined. In the end, evaluating the experiences of teachers, trainers, and evaluators is essential to identify human resources.
	Determine Potential Delivery System	Showing all the system delivery options and estimating each phase's cost in the ADDIE model. All available delivery methods should be listed, i.e. (physical courses, digital courses, or hybrid courses). Then, analyze each choice and calculate the delivery time and cost required.
	Compose a Project Management Plan	Documenting the project's overall plan from a project management perspective. The project plan covers the team members participating in each phase of the project and their experiences. It also contains what will include and exclude in the project, as well as a schedule that shows the tasks duration, dependence, and a clear time plan for the project. Triggers the creation of the final report and updates it throughout subsequent phases. It should cover what happened in the project to evaluate its progress and make improvements in the following projects.
Design phase	Conduct a Task Inventory	Defining the tasks to achieve the proposal statement. Sequential actions occur to create a tree diagram in that the root is the purpose statement, and the leaves are the performance tasks. First, clarify the purpose statement with all the team members and stakeholders. Second, decomposing the purpose statement into instructional goals. Third, specify the performance tasks in order to achieve the instructional goals. Fourth, identifying the knowledge and skills needed to create each performance task
	Compose Performance Objectives	Defining the prerequisite knowledge required for the students to take a specific performance task, Create acceptance criteria to evaluate the student's knowledge and skill. The performance objectives should answer the following three questions, what are the student activities during the task? How will the student gain the expected performance? Furthermore, what are the acceptable criteria for the performances?
	Generate testing Strategies	Defining items used to test the performance tasks and performance objectives to guarantee the achievement of the performance criteria. The goal of the learning process is to increase student's performance. Because of that, measuring the student's performance after each performance task is essential.
	Calculate Return on Investment (ROI)	Common process in each project that answers whether it is relative to the investment cost. In the ADDIE approach, it occurs as the last process in the design phase.
Develop phase	Generate Content Instruction Strategy	Constructing the learning content based on the design brief (the output of the design phase). Effective learning content should have three components, a beginning component that shows the prerequisite knowledge required, clear motivation, and the achieved goals after learning. The middle component contains the content that, students have to learn in different forums, i.e., presentations, open discussions, or exercises. The end component makes a summary of the content and a review of the achieved goals. Additionally, Instructional strategies such as understanding students' motivation, style, and rate of learning, ensuring course material's logical order, and engaging the material with media material have a significant impact on the development of the content.
	Select or Develop Media	Facilitating learning by developing or selecting media material that transparently explains the performance tasks. The media content can be PowerPoint slides, videos, or storyboards.
	Develop Guidance for the Student and Teachers	Two consecutive processes, which define the instructions and guidelines for both students and teachers In case of the students, the guidance explains the best strategy to learn the content material. In case of the teachers, the guidance is related to the best strategy to clarify the learning content. Showing students/teachers what is expected of them has a significant impact on their performance. However, deigning the table of contents and appendix and decomposing content into the beginning, middle, and end will increase learning efficiency.
	Conduct Formative Revisions	Evaluating the generated learning material in order to evaluate/improve designed content and decrease the performance gap. The evaluation occurs by gathering data about students learning curves while using the learning material. The data is collected by pre- / post-tests, observation, or questionnaire. The formative evaluation is done in three phases. First, one-to-one trials, tend to remove the apparent mistakes. Second, the small group trial, get feedback to determine the effectiveness of the learning material. Third, the Field trial, decided whether to start the implementation phase or improve the material.
	Conduct a Pilot Test	The pilot test results support the final decision before going to the implementation phase. The participating students should represent the student for whom the course was created. The pilot test results should be collected and analyzed in order to check each specific learning content and instruction strategy. evaluation of the results and generating a pilot test result is the responsibility of the evaluation team. The customer should make the final decision based on the pilot test results.
Implementation phase	Prepare the Teacher	Training the teachers to use the developed learning content, learning resources, and instructional strategies to decrease the student's performance gap and increase learning efficiency. The course admin/manager (University or school) defines the potential trainers for the course based on the trainer's experiences and background, then schedules a train-to-trainer course that explains to the potential teachers how to facilitate the learning process.
	Prepare the Student	Motivate the students for the new course. A learning plan should be clarified during this process which includes understanding the student's learning style, defining the prerequisite knowledge required, creating a schedule for the sessions of the course and the number of students per class, and pre-communication with the students in order to provide them with all information. Developing methods to Track and record the student's performance during the course. The tracking process can be done through exams, exercises, or observation.
Evaluation phase	Determine evaluation criteria	During the design phase, the standard of learning quality was defined. The objective of this process is to check whether the learning resource meets its standards by identifying the following three components. Teachers measure the student's reactions and perceptions, judge the student learning curves and their ability to understand the learning resources, and administered supervisors check whether the learning process closes the performance gap.
	Select evaluation tools and conduct evaluations	The last two steps in the evaluation phase and the ADDIE model. Select evaluation tools process identifies the tasks that will be evaluated and determines the suitable tools for them. Conduct evaluation refers to the process of carrying out the evaluation, which should be done by the implementation team rather than the ID team.

TABLE II

DESCRIPTION OF THE CONCRETE PROCESSES WITHIN THE ADDIE MODEL

Phase	Process	Description
Requirement phase	Requirements elicitation and analysis	Understanding stakeholder work and how a new system may assist it is the goal of the requirements elicitation process. In order to learn more about the application domain, workflow, required system features and performance, hardware limitations, and other topics, software developers collaborate with stakeholders. This process can be challenging because stakeholders might not know what they want, struggle to articulate their requirements, make unreasonable requests, express varied and contradictory requirements, are influenced by political factors, and have changing requirements due to a dynamic business and economic environment.
	Requirements specification	The process of writing user and system requirements is known as a requirements specification. While having clear, unambiguous, comprehensive, and consistent standards is ideal, this can be difficult to achieve in practice due to ambiguity and conflicts. Although system requirements can also be expressed in natural language or symbols such as forms and graphical or mathematical system models, user requirements are usually written naturally together with diagrams and tables. User requirements should state only the system's external behavior, not include architectural or design specifics, and interpret functional and non-functional requirements in a way that non-technical users can understand.
	Requirements validation	Requirements validation is the process of ensuring that requirements define the system the customer needs. This is important because errors in the requirements document can lead to costly rework, either during development or after the system goes live. Requirements documents are subject to different types of checks during validation, including validity checks to ensure that they reflect real user needs, consistency checks to avoid claims conflicting requirements, completeness checks to ensure that all functions and constraints are defined, reality checks to ensure the project can be completed on time and within budget, and finally, verifiability checks to ensure that requirements can be tested and proven in the delivered system. This ensures that the finished system will meet the needs of the customer.
	Requirements management	The process of deciding how to manage a set of change requirements is called requirements management. Several issues must be addressed during the planning phase, such as defining each requirement individually for cross-checking and traceability, change management processes for impact assessment, shift costs, and a traceability policy to define relationships between requirements. This requires design support systems and tools to manage large amounts of information. Automation support, which can be in the form of standard spreadsheets, databases, or specialized tools, is needed to manage requirements. Software technologies that provide secure storage of requirements, change management, and traceability management should be selected during the planning phase. Some technologies can also find correlations between criteria using natural language processing techniques.
Design phase	System modeling	The practice of creating abstract representations of a system using various mathematical or graphical notations is known as system modeling. While formal modeling uses mathematical notation, graphical modeling often uses UML. Models are used in the requirements engineering process to create specific requirements, in the system design phase to communicate the system with engineers, and in the post-implementation phase to document the system's functionality. Models of the current system can be used to understand its advantages and disadvantages. In contrast, models of the future system can be used to clarify requirements, support equipment design, and implement changes.
	Architecture design	The process of comprehending and designing the overall structure of a software system. It serves as a critical link between design and requirements engineering by identifying the main structural components of a system and their relationships. The result of architectural design is an architectural model that describes the system as a collection of communicating components.
	Object-oriented design with UML	The process of creating the structure and interactions of the system's objects is known as object-oriented system design. These objects have a private representation that is inaccessible from the outside, maintain their local state, and conduct actions on it. When an object-oriented design is implemented as a program, classes of objects and their relationships are created and used to produce objects dynamically. Objects can be understood and changed as distinct entities containing data, as well as actions for modifying that data. Providing an explicit mapping between actual entities in the real world and the system's controlling objects enhances the design's comprehension and maintainability.
Implementation phase	System implementation	Software engineering is a process that encompasses all aspects of software development, from initial requirements to system maintenance and management. System implementation is a critical stage in this process in which an executable version of the software is created. This may entail writing programs in high or low-level languages or adapting off-the-shelf systems to meet an organization's specific needs.
Testing phase	Development testing	Development testing is the process by which the system development team tests the software. Software is usually tested by the programmers who created it, but some development processes use programmer-tester pairs or separate testing groups within the development team. Unit testing, where a single program unit or class of objects is tested; component testing, where multiple individual units are integrated to form a composite component and tested on component interfaces; system testing that integrates some or all of the system's components and tests the system as a whole with an emphasis on testing the interactions of the components.
	Release testing	Release testing is the process of evaluating a particular version of a system for use by parties other than the development team, such as customers or other groups working on that system. It differs from system testing because the development team does not perform it but performs it during system development. The emphasis is on ensuring that the system meets requirements and is suitable for the customer's use rather than finding defects. The primary purpose of release testing is to demonstrate that the system meets its intended functionality, performance, and reliability requirements and will not fail under typical usage conditions. This will convince the supplier or customer that the system is suitable for release as a product or service.
	User testing	User testing is the process where users or customers provide feedback and suggestions for system testing. This may include formal evaluation of systems ordered from third-party suppliers to test new software products. This is critical even after complete system and release testing because the user's work environment can affect system reliability, performance, usability, and robustness. There are three types of user testing: alpha testing is when a small group of users works closely with the development team to test an early version, beta testing is when a larger number of users are given access to a version of the software for evaluation, and acceptance testing, where the customer tests the system to determine whether it is ready for acceptance and deployment in the customer's environment.
Operation phase	Maintenance	Modifying the system after its delivery is called software maintenance. This is often used when two different development teams work on custom software before and after delivery. Simple coding problems can be fixed, design errors can be corrected more thoroughly, and software can be significantly modified to correct specification errors or meet new requirements. There are three types of software maintenance: bug fixes, environment adaptation, and feature additions.
	Updating	Software updating entails modifying the software to reduce the performance hit brought on by the modification. This calls for software modifications that improve its structure, decrease its complexity, or simplify it. It can be applied to any development process, not just object-oriented development, and should emphasize enhancing the software more than introducing new features.

TABLE III

DESCRIPTION OF THE CONCRETE PROCESSES WITHIN THE WATERFALL MODEL