GANPAT UNIVERSITY

FACULTY OF DIPLOMA ENGINEERING

Programme	Diploma in Computer Engineering/ Information Technology				
Semester	IV		Version	1.0.0.0	
Effective from Academic Year		2026-27	Effective for the batch Admitted in		JULY 2025
Course Code	1CEIT4103	Course Name	Software Engineering		

I. TEACHING-LEARNING AND ASSESSMENT SCHEME

Course Type	Course code	Course Title	Teaching & Learning Scheme									Examination Scheme							
			Credit				Actual Contact Hrs/week			SLH	Total Learning Hrs/Week	TH			PR			SLA	Total
			CL	TL	LL	Total	CL	TL	LL			CE	SEE	Total	CE	SEE	Total		
DSC	1CEIT4103	Software Engineering	3	0	1	4	3	-	2	7	7	40	60	100	30	20	50	20	170

Abbreviation:	**CL-** Classroom Learning	**TL -** Tutorial Learning	**LL -** Laboratory Learning
	SLH - Self Learning Hours	**SLA -** Self Learning Assessment	**CE -** Continuous Evaluation
	SEE – Semester End Examination		

II. PRE-REQUISITES

Basic knowledge of computer fundamentals and object-oriented programming concepts is desirable.

III. INDUSTRY /EMPLOYER EXPECTED OUTCOMES

Ability to analyse, design, test, and manage maintainable software systems using standard software engineering practices for academic projects and entry-level industry roles.

IV. COURSE LEARNING OUTCOMES

At the end of the course, students will be able to achieve the following course learning outcomes:

CO1. Analyse & Compare Software Process Models.
CO2 Analyse the role and importance of requirement analysis in software development.
CO3. Design software solutions using SRS, DFDs, and object-oriented UML diagrams.
CO4. Prepare a software development plan using appropriate project scheduling methods.
CO5. Design and apply test cases to evaluate software functionalities.

V. THEORY LEARNING OUTCOMES AND ALIGNED COURSE CONTENT:

Name of Unit	Theory Learning outcomes (TLO's) aligned to CO's	Learning Content mapped with Theory Learning outcomes (TLO's)&CO's	Marks	Hours
Unit-1 **Software Development Approaches**	**TLO 1.1** Understand basics and importance of Software **TLO 1.2** Describe the characteristics of the software. **TLO 1.3** State the software applications. **TLO 1.4** Outline software engineering. **TLO 1.5** Illustrate the working of the Waterfall model **TLO 1.6** Identify the working principle of the Incremental Process Model. **TLO 1.7** Understand the	**1.1** Introduction of Software **1.2** Software Characteristics **1.3** Software Applications **1.4** Software Engineering: A layered Approach Generic Framework activity and umbrella activity **1.5** Software Development Models, Waterfall model **1.6** Incremental process model **1.7** Prototype model **1.8** Spiral model **1.9** RAD model **1.10** Agile Approach - Principles	**10**	**12**

	working of the Prototype Model. **TLO 1.8** Interpret the working principle of the Spiral Model. **TLO 1.9** Describe the working principle of the RAD Model. **TLO 1.10** Summarize the principles of Agile development	of Agile, Agile Frameworks - Scrum, Extreme Programming (XP)		
Unit-2 **Software Requirement Analysis**	**TLO 2.1** Describe requirement gathering and analysis. **TLO 2.2** Classify the types of requirements **TLO 2.3** Describe SRS, formal specification techniques. **TLO 2.4** Interpret the characteristics of a good SRS.	**2.1** Requirement Gathering and Analysis, Techniques for Requirement Gathering **2.2** Types of Requirements **2.3** Software Requirement Specification (SRS) **2.4** Ccharacteristics of a Good SRS	**12**	**06**
Unit-3 **Software Design and Modelling**	**TLO 3.1** Understand the characteristics of good software. **TLO 3.2** Describe cohesion and coupling. **TLO 3.3** Develop and interpret DFD Diagram **TLO 3.4** Explain Use Case Diagram. **TLO 3.5** Describe Class Diagram. **TLO 3.6** Represent Sequence Diagram. **TLO 3.7** Describe Activity Diagram. **TLO 3.8** Prepare Component Diagram.	**3.1** Characteristics of a Good software design **3.2** Classification of cohesion and classification of coupling. **3.3** Function Oriented Software Design -Data flow diagram (DFD), Context level DFD and Level -1 DFD **3.4** Object-Oriented Modelling with UML – Use case diagram **3.5** Class Diagram **3.6** Sequence Diagram **3.7** Activity Diagram **3.8** Component Diagram	**14**	**12**
Unit-4 **Software Project Estimation and Scheduling**	**TLO 4.1** Summarize the responsibility of Software Project manager. **TLO 4.2** Identify and explain the project metrics. **TLO 4.3** Describe the project estimation using different methods. **TLO 4.4** Interpret the importance of project scheduling**.** **TLO 4.5** Understand the concept and importance of risk management in software projects. **TLO 4.6** Prepare a risk	**4.1** Responsibility of software project Manager **4.2** Metrics for Size Estimation - Line of Code, Function Points **4.3** Project Estimation Techniques - COCOMO model **4.4** Project Scheduling - Gantt Chart, Sprint burndown chart for agile model **4.5** Risk Management, Risk Identification, Risk Assessment **4.6** Risk mitigation and planning **4.7** Risk Monitoring & Control	**14**	**09**

	management plan including mitigation and contingency actions. **TLO 4.7** Identify the need for continuous risk monitoring in software projects.			
Unit-5 **Software Coding and Testing**	**TLO 5.1** Identify the different types of coding methods. **TLO 5.2** Classify code review techniques. **TLO 5.3** Summarize software documentation types. **TLO 5.4** Understand the concept of testing. **TLO 5.5** Describe the Black box software testing. **TLO 5.6** Illustrate the concept of White box software testing. **TLO 5.7** Describe the Alpha & Beta Testing in software testing. **TLO 5.8** Explain the Unit testing and Integration testing. **TLO 5.9** Apply test case templates to validate software functionality systematically.	**5.1** Coding standards and guidelines **5.2** Code Walkthrough and Code Inspection **5.3** Internal Documentation and External Documentation **5.4** Testing Fundamentals **5.5** Black box testing **5.6** White box testing **5.7** Alpha & Beta Testing **5.8** Unit testing & Integration testing **5.9** Test Documentation – test case template	**10**	**06**

VI. LABORATORY LEARNING OUTCOME AND ALIGNED PRACTICAL			
SR. NO	**PRACTICAL/LABORATORY LEARNING OUTCOME(LLO)**	**PRACTICAL TITLES**	**RELEVANT COs**
1	**LLO 1.1** Analyse the advantages and limitations of different software development models.	Demonstrate different software development models using suitable diagrams	CO1
2	**LLO 1.2** Define the scope and boundaries of a software project.	Prepare a problem statement that clearly defines the project title and specifies the bounded scope of the project.	CO1
3	**LLO 2.1** Identify application-specific functional and non-functional requirements.	Gather application specific requirements- Requirement gathering.	CO2
4	**LLO 2.2** Prepare a broad SRS for a selected project.	Prepare broad SRS (software requirement specification) for the above selected project.	CO2
5	**LLO 3.1** Develop Data Flow Diagrams (DFDs) at different levels.	Develop data designs using DFDs and E-R diagram.	CO3
6	**LLO 3.2** Develop use cases and use case diagrams	Prepare use-cases and draw use case diagram.	CO3
7	**LLO 3.3** Understand the purpose of class diagrams in object-oriented design.	Develop a class diagram for selected project.	CO3
8	**LLO 3.4** Understand the purpose of	Develop Sequence diagram for selected	CO3

	sequence diagrams in UML.	project.	
9	**LLO 3.5** Understand the concept and purpose of activity diagrams in software development.	Develop the activity diagram to represent flow from one activity to another for software development.	CO3
10	**LLO 3.6** Understand the concept and purpose of component diagrams in software development.	Develop Component diagram for selected project.	CO3
11	**LLO 4.1** Understand the importance of software project planning in software development.	To prepare a detailed Software Project Plan including schedule, milestones, resources, and Gantt Chart.	CO4
12	**LLO 4.2** Identify and analyze risks in software projects.	To identify and analyze software project risks and prepare a risk mitigation plan.	CO4
13	**LLO 4.3** Evaluate project size to support effort and cost estimation.	Evaluate size of the project using Function point metric for the assigned project.	CO4
14	**LLO 5.1** Design different types of test cases	Prepare various test case for selected project.	CO5
15	**LLO 5.2** Study about testing tools	To design and perform different levels of software testing and analyze test results for the selected project.	CO5

VII. SUGGESTED MICRO PROJECT/ASSIGNMENTS/ACTIVITIES FOR SELF LEARNING/SKILL DEVELOPMENT (SELF LEARNING)

Micro Projects (Mini Applications / Use Cases)

- Design and document a mini software application by performing requirement analysis, SDLC and Agile modelling, UML and data design, project estimation, scheduling, risk management, and test case preparation.

Self-Learning / Skill Building Activities

1. Problem Statement and Project Scope Definition.
2. Requirement Gathering and Broad SRS Preparation
3. Study of Software Development Life Cycle (SDLC) Models
4. Agile Methodology – Scrum and XP Basics
5. Use Case Preparation and Use Case Diagram
6. UML Diagrams: Class, Sequence, and Activity Diagrams
7. Data Design using DFDs and E-R Diagrams
8. Software Size Estimation using Function Point Metric
9. Project Scheduling using Gantt Chart and Sprint Burn Down Chart
10. Risk Identification, Mitigation, and Monitoring
11. Test Documentation – Test Case Template Preparation

VIII. LIST OF INSTRUMENTS / EQUIPMENT / TRAINER BOARD

1	Computer system with basic configuration, office productivity tools, and UML/DFD diagramming software.
2	Internet connectivity for accessing online tools
3	Diagramming tools for UML and DFD diagrams

IX. LIST OF REFERENCE BOOKS			
Sr.No	**Title**	**Author**	**Publication**
1	Software Engineering: A Practitioner's Approach	Roger S. Pressman	Tata McGraw Hill
2	Fundamentals of Software Engineering	Rajib Mall	PHI
3	Object Oriented Modelling and design with UML	Michael R Blaha and James R Rambaugh	Pearson Prentice Hall

X. LINK OF LEARNING WEB RESOURCE	
1	https://nptel.ac.in/courses/106101061
2	https://www.tutorialspoint.com/software_engineering/
3	https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development
4	https://nptel.ac.in/courses/106105087

XI. SUGGESTED WEIGHTAGE TO LEARNING EFFORTS & ASSESSMENT PURPOSE							
Unit	**Unit Title**	**Aligned COs**	**Learning Hours**	**R-Level**	**U-Level**	**A-Level**	**Total Marks**
1	Software Development Approaches	CO1	12	3	3	4	10
2	Software Requirement Analysis	CO2	06	3	4	5	12
3	Software Design and Modelling	CO3	12	3	5	4	14
4	Software Project Estimation & Scheduling	CO4	09	4	6	4	14
5	Software Coding and Testing	CO5	06	4	5	5	10
	Grand Total		**45**	**17**	**23**	**20**	**60**

XIII. COs AND POs AND PSOs MAPPING										
Course Outcome (Cos)	**Programme Outcomes (POs)**							**Programme Specific Outcomes (PSOs)**		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PSO1	PSO2	PSO3
CO1	3	2	1	1	0	1	0	3	1	0
CO2	3	2	3	1	0	0	0	3	1	1
CO3	3	2	3	2	0	1	0	3	1	1
CO4	3	3	3	2	1	1	0	3	1	1
CO5	2	3	3	2	0	1	0	3	1	1

Legends: - *3-High; 2-Moderate/Medium; 1-Slight/Low; 0-None*