Anekant Education Society's

Tuljaram Chaturchand College, Baramati

(Empowered Autonomous)

Three Year B.Sc. Degree Program in Computer Science

(Faculty of Science & Technology)

CBCS Syllabus (2024 Pattern)

T.Y. B.Sc. (Computer Science) Sem- V (2024 Pattern)

For Department of Computer Science

Tuljaram Chaturchand College of Arts, Science & Commerce, Baramati

Choice Based Credit System Syllabus (2024Pattern)

(As Per NEP2020)

To be implemented from Academic Year 2026-2027

Title of the Programme: T.Y.B.Sc.(Computer Science) 2024 Pattern

Preamble

AES's Tuljaram Chaturchand College has made the decision to change the syllabus of various faculties from June, 2023 by incorporating the guidelines and provisions outlined in the National Education Policy (NEP), 2020. The NEP envisions making education more holistic and effective and to lay emphasis on the integration of general (academic) education, vocational education and experiential learning. The NEP introduces holistic and multidisciplinary education that would help to develop intellectual, scientific, social, physical, emotional, ethical and moral capacities of the students. The NEP 2020 envisages flexible curricular structures and learning based outcome approach for the development of the students. By establishing a nationally accepted and internationally comparable credit structure and courses framework, the NEP2020 aims to promote educational excellence, facilitate seamless academic mobility, and enhance the global competitiveness of Indian students. It fosters a system where educational achievements can be recognized and valued not only within the country but also in the international arena, expanding opportunities and opening doors for students to pursue their aspirations on a global scale.

In response to the rapid advancements in science and technology and the evolving approaches in various domains of Computer Science and related subjects, the Board of Studies in Computer Science at Tuljaram Chaturchand College, Baramati - Pune, has developed the Credit, Course Structure of T.Y.B.Sc.(Computer Science) Sem- V, VI (2024 Pattern) and curriculum for the Fifth semester of T.Y.B.Sc.(Computer Science) (2024 Pattern), which goes beyond traditional academic boundaries. The syllabus is aligned with the NEP 2020 guidelines to ensure that students receive an education that prepares them for the challenges and opportunities of the 21st century. This syllabus has been designed under the framework of the Choice Based Credit System (CBCS), taking into consideration the guidelines set forth by the National Education Policy (NEP) 2020, LOCF (UGC), NCrF, NHEQF, Prof. R.D. Kulkarni's Report, Government of Maharashtra's General Resolution dated 20thApril and 16th May 2023, and the Circular issued by SPPU, Pune on 31st May 2023.

A degree in Computer Science subject equips students with the knowledge and skills necessary for a diverse range of fulfilling career paths. Career in Computer Science is

considered one of the most high-paying jobs and is full of opportunities; particularly when India's prowess in information technology industry is recognized across the globe. The pool of talented computer professionals working in IT companies of the USA, Canada and other countries shows that IT can take a person to higher levels. Numerous IT companies from India employ huge number of computer professionals in their Indian and overseas offices. Students who are interested in programming**,** software development, and have good analytical and reasoning skills may pursue this course. Job opportunities are available for Graduates and Post Graduates in Government as well as Private sector. Graduates may take up the following job posts-Software Engineer, Software Tester, Data Analyst, Project Manager, Network Administrator, Database Administrator and Application Developer. Overall, revising the Computer Science syllabus in accordance with the NEP 2020 ensures that students receive an education that is relevant, comprehensive, and prepares them to navigate the dynamic and interconnected world of today. It equips them with the knowledge, skills, and competencies needed to contribute meaningfully to society and pursue their academic and professional goals in a rapidly changing global landscape.

Programme Specific Outcomes (PSOs)

B.Sc. (Computer Science)

PSO1: Apply fundamental principles and methods of Computer Science to a wide range of applications.

PSO2: Design, correctly implement and document solutions to significant computational problems.

PSO3: Impart an understanding of the basics of our discipline.

PSO4: Prepare for continued professional development.

PSO5: Understand the impact of the IT analyst solutions in societal and environmental contexts, and demonstrate the knowledge and need for sustainable development.

PSO6: Develop proficiency in the practice of computing.

PSO7: Develop the capacity to study and research independently that will help to develop skills for transition to employment in hardware/software companies.

Anekant Education Society's

Tuljaram Chaturchand College, Baramati

(Empowered Autonomous)

Board of Studies(BOS) in Computer Science

From 2025-26 to 2027-28

Sr.No.	Name	Designation
1.	Dr. Upendra D. Choudhari	Chairman
2.	Dr. Vilas V. Kardile	Member
3.	Mr. Abhijeet D. Mankar	Member
4.	Mrs. Prajakta P. Kulkarni	Member
5.	Mrs. Asmita A. Bhagat	Member
6.	Mr. Rahul A. Shah	Member
7.	Mr. Purushottam S. Dixit	Member
8.	Dr. Shashikant C. Nakate	Member
9.	Mrs. Poornima C. Swami	Member
10.	Mrs. Komal M. Theurkar	Member
11.	Mr. Swapnil Chemte	Member
12.	Mrs. Lata S. Jadhav	Member
13.	Mrs. Jyostna P. Gharge	Member
14.	Ms. Vaishnavi K. Shivarkar	Member
15.	Mrs. Vrushali Y. Shirkande	Member
16.	Dr. Manisha Bharambe	Vice-Chancellor Nominee
17.	Dr. Sudhakar Bhoite	Expert from other University
18.	Dr. Ulhas S. Patki	Expert from other University
19.	Mr. Yadav Preetam	Representative from Industry/Corporate Sector
20.	Mr. Bhaskar Ranaware	Member of the college Alumni
21.	Ms. Sakshi Vargar	Student Representative
22.	Mr. Adesh Jagtap	Student Representative

Credit & Course Structure for T.Y.B.Sc.(Computer Science)(2024Pattern) NcrF Credit Level : 5.5

As per NEP-2020

Sem	Course Type	Course Code	Course Title	Course Types	Credits
V	Major Mandatory	COS-301-MRM	Operating Systems	Theory	02
	Major Mandatory	COS-302- MRM	Theoretical Computer Science	Theory	02
	Major Mandatory	COS-303- MRM	Computer Networking	Theory	02
	Major Mandatory	COS-304- MRM	Object Oriented Software Engineering	Theory	02
	Major Mandatory	COS-305- MRM	Lab Course Based on Advanced Java	Practical	02
	Major Mandatory	COS-306- MRM	Lab Course Based on COS-301-MRM	Practical	02
	Major Elective (MJE)	COS-307-MJE(A)	Core PHP	Theory (Any One)	02
	Major Elective (MJE)	COS-307-MJE(B)	Blockchain Technology		
	Major Elective (MJE)	COS-308-MJE(A)	Lab Course based on COS-307-MJE(A)	Practical (Any One)	02
	Major Elective (MJE)	COS-308-MJE(B)	Lab Course based on COS-307-MJE(B)		
	On Job Training (OJT)	COS-385-OJT	On Job Training	Practical	04
	Minor	COS-310-MN(A)	Predictive Analytics	Theory (Any One)	02
		COS-310-MN(B)	Linear Algebra		
		COS-310-MN(C)	Architecture of 8051Microcontroller and it's programming		
	Total Credits Semester – V				**22 (T=12 P=10)**
VI	Major Mandatory	COS-351-MRM	Advanced Operating Systems	Theory	02
	Major Mandatory	COS-352- MRM	Compiler Construction	Theory	02
	Major Mandatory	COS-353- MRM	Upper layers of Network and Network Security	Theory	02
	Major Mandatory	COS-354- MRM	Software Architecture and testing tools	Theory	02
	Major Mandatory	COS-355- MRM	Lab Course based on Android Programming	Practical	02
	Major Mandatory	COS-356- MRM	Lab Course Based COS-351-MRM & COS-353- MRM	Practical	02
	Major Elective (MJE)	COS-357-MJE(A)	Advanced PHP	Theory (Any One)	02
	Major Elective (MJE)	COS-357-MJE(B)	IoT		
	Major Elective (MJE)	COS-358-MJE(A)	Lab Course based on COS-357-MJE(A)	Practical (Any One)	02
	Major Elective (MJE)	COS-358-MJE(B)	Lab Course based on COS-357-MJE(B)		
	Vocational Skill Course (VSC)	COS-359- VSC	Python Programming	Theory	02
	Vocational Skill Course (VSC)	COS-360- VSC	Lab Course Based on COS-359- VSC	Practical	02
	Field Project (FP)	COS-361-FP	Field Project	Practical	02
	Total Credits Semester – VI				**22**
	Grand Total Semester- V + Semester- VI				**44**

T.Y.B.Sc. (Computer Science)

Semester- V

Credit Structure & Syllabus

2024 Pattern

(Academic Year 2026-2027, Empowered Autonomous)

Course Structure for T. Y. B. Sc. (Computer Science) Sem-V & VI (2024 Pattern)
Subject: Computer Science
To be implemented from Academic Year 2026-2027

Sem	Course Code	Title of the Course	No. of Credits	Exam	Marks
V	COS-301-MRM	Operating Systems	2	I / E	30 + 20
	COS-302- MRM	Theoretical Computer Science	2	I / E	30 + 20
	COS-303- MRM	Computer Networking	2	I / E	30 + 20
	COS-304- MRM	Object Oriented Software Engineering	2	I / E	30 + 20
	COS-305- MRM	Lab Course Based on Advanced Java	2	I / E	30 + 20
	COS-306- MRM	Lab Course Based on COS-301-MRM	2	I / E	30 + 20
	COS-307-MJE(A)	Core PHP	2	I / E	30 + 20
	COS-307-MJE(B)	Blockchain Technology		I / E	30 + 20
	COS-308-MJE(A)	Lab Course based on COS-307-MJE(A)	2	I / E	30 + 20
	COS-308-MJE(B)	Lab Course based on COS-307-MJE(B)			
	COS-385-OJT	On Job Training	4	I / E	60 + 40
	COS-310-MN(A)	Predictive Analytics	2	I / E	30 + 20
	COS-310-MN(B)	Linear Algebra			
	COS-310-MN(C)	Architecture of 8051Microcontroller and it's programming			
VI	COS-351-MRM	Advanced Operating Systems	2	I / E	30 + 20
	COS-352- MRM	Compiler Construction	2	I / E	30 + 20
	COS-353- MRM	Upper layers of Network and Network Security	2	I / E	30 + 20
	COS-354- MRM	Software Architecture and testing tools	2	I / E	30 + 20
	COS-355- MRM	Lab Course based on Android Programming	2	I / E	30 + 20
	COS-356- MRM	Lab Course Based COS-351-MRM & COS-353- MRM	2	I / E	30 + 20
	COS-357-MJE(A)	Advanced PHP	2	I / E	30 + 20
	COS-357-MJE(B)	IoT			
	COS-358-MJE(A)	Lab Course based on COS-357-MJE(A)	2	I / E	30 + 20
	COS-358-MJE(B)	Lab Course based on COS-357-MJE(B)	2	I / E	30 + 20
	COS-359- VSC	Python Programming	2	I / E	30 + 20
	COS-360- VSC	Lab Course Based on COS-359- VSC	2	I / E	30 + 20
	COS-361-FP	Field Project	2	I / E	30 + 20

SYLLABUS (CBCS as per NEP 2020) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern (w. e. from AY 2026-27)

Name of the Programme : B.Sc. Computer Science
Program Code : USCOS
Class : T.Y. B.Sc. (Comp. Sci.)
Semester : V
Course Type : Major (TH)
Course Name : Operating Systems
Course Code : COS-301-MRM
No. of Lectures : 30
No. of Credits : 2

Prerequisites:

- Basic Knowledge of Computer

Course Objectives:

- To understand Complexity of Operating system as a software. .
- To understand design issues related to process management and various related algorithms
- To understand design issues related to Thread concepts

Course Outcome:

CO1: Learn the working of various operating system modules.
CO2: Able to understand the concept of process scheduling.
CO3: Learn system calls and their role in interacting with the operating system.
CO4: Study the role of the operating system in managing memory.
CO5: Learn the Multithreading Concepts
CO6: Understand how process synchronization is done
CO7: Learn concept of deadlock management.

UNIT No.	Chapter Name with Topics	No. of Lectures Required
UNIT- I	**Operating System as System Software** 1.1 What Operating Systems Do – User View, System View, Defining OS 1.2 Computer System Architecture – Single processor system, Multiprocessor systems, Clustered Systems 1.3 Operating System Operations – Dual mode operation, Timer 1.4 Process Management 1.5 Memory Management 1.6 Operating System Services 1.7 User Operating-System Interface – Command interpreter, GUI 1.8 System Calls and Types: -Process control, File management, Device management, Information maintenance, Communication, Protection	**06**
UNIT- II	**Process Management and Scheduling** 2.1 Process Concept – The process, Process states, Process control block.	**10**

	2.2 Process Scheduling – Scheduling queues, Schedulers, context switch 2.3 Operations on Process – Process creation with program using fork (), Process termination 2.4 Inter-process Communication – Shared memory system, Message passing systems. 2.5 Basic Concept – CPU-I/O burst cycle, CPU scheduler, Preemptive scheduling, Dispatcher 2.6 Scheduling Criteria 2.7 Scheduling Algorithms – FCFS, SJF, Priority scheduling, Round-robin scheduling, Multiple queue scheduling, Multilevel feedback queue scheduling. 2.8 Multithreaded Programming 2.9 Multithreading Models 6.6 Thread Scheduling	
UNIT-III	**Multithreaded Programming** 3.1 Overview 3.2 Multithreading Model 3.3 Thread Libraries P-Tread, Java Thread 3.4 Thread Life Cycle	**04**
UNIT-IV	**Process Synchronization and Deadlock** 4.1 Background 4.2 Critical Section Problem 4.3 Semaphores: Usage, Implementation 4.4 Classic Problems of Synchronization – The bounded buffer problem, The reader writer problem, The dining philosopher problem 4.5 System model 4.6 Deadlock Characterization – Necessary conditions, Resource allocation graph 4.7 Deadlock Prevention 4.8Deadlock Avoidance - Safe state, Resource allocation graph algorithm, Banker's Algorithm 4.9Deadlock Detection 4.10 Recovery from Deadlock – Process termination, Resource Preemption	**10**

Reference Books

- Siberchatz, Galvin, Gagne Operating System Concepts - (8th Edition).
- Pabitra Pal Choudhary Operating Systems : Principles and Design – (PHI Learning Private Limited)
- Maurice J. Bach.The Design of the UNIX Operating System, PHI
- Mukesh Singhal, Niranjan G Shivarat Advanced Concepts in Operating Systems

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	2	2	2	2	1	1	2	3	1	1	2	1
CO2	3	3	2	3	3	2	1	2	3	1	1	2	1
CO3	3	3	2	3	3	2	1	2	3	1	1	2	1
CO4	3	3	2	3	3	1	1	2	3	1	1	2	1
CO5	2	3	3	3	3	2	1	2	3	1	1	2	1
CO6	3	3	3	3	3	2	1	2	3	1	1	2	1
CO7	3	3	2	3	3	2	1	2	3	1	1	2	1

Mapping of CO with PO(3: Strongly Related, 2: Moderately Related, 1: Partially Related)
Justification:

PO1 with All COs

CO1: Understanding OS modules directly contributes to strong foundational knowledge.
CO2: Process scheduling forms a core concept in OS knowledge.
CO3: System calls are crucial for understanding OS-user interaction.
CO4: Memory management is a vital theoretical concept.
CO5: Multithreading is significant but more applied in nature.
CO6: Synchronization is essential for OS core understanding.
CO7: Deadlock management is a key concept in OS theory.

PO2 with All COs:

CO1 Understanding modules contributes moderately to practical OS use.
CO2 Process scheduling is practically implemented and evaluated.
CO3 System calls are widely used in OS programming.
CO4 Hands-on memory management is key in labs and development.
CO5 Multithreading is heavily used in practical scenarios.
CO6 Synchronization techniques are important for concurrency control.
CO7 Practical strategies for avoiding deadlocks are crucial in system design.

PO3 with All COs

CO1 Encourages system-level thinking useful in tech entrepreneurship.
CO2 Process optimization can inspire performance-driven innovation.
CO3 System-level APIs can lead to low-level system innovation.
CO4 Efficient memory strategies can be a business differentiator.
CO5 Multithreading enables scalable, high-performance applications.
CO6 Safe synchronization is critical in commercial software.
CO7 Deadlock resolution plays a role in designing reliable systems.

PO4 with All COs

CO1 Introduces specialized OS components.
CO2 Develops algorithmic scheduling skills.
CO3 Enhances ability to interact with system-level APIs.
CO4 Builds competency in memory management techniques.
CO5 Develops skills in handling concurrent programming.
CO6 Strengthens ability in synchronization methods.
CO7 Builds deadlock handling skills vital for system safety.

PO5 with All COs

CO1 Moderate application of module concepts in solving OS problems.
CO2 Analytical reasoning in selecting appropriate scheduling algorithms.
CO3 Requires applying system calls in various scenarios.
CO4 Involves solving memory-related problems.
CO5 Needs problem-solving in thread coordination.
CO6 Analytical handling of synchronization issues.
CO7 Deadlock detection and resolution demands strong reasoning.

PO with All COs

CO1 Limited relevance to communication.
CO2 Collaborative problem-solving in labs.
CO3 Encourages communication through coding and debugging.
CO4 Less focused on group collaboration.
CO5 Possible collaboration in multithreaded projects.
CO6 Requires communication in team-based synchronization tasks.
CO7 Discussion and debugging deadlocks involve teamwork.

PO7 with All COs

CO1 Offers theoretical base but limited research.
CO2 Not typically research-oriented.
CO3 Some scope in systems research.
CO4 Limited direct research applications.
CO5 Multithreading has potential research aspects.
CO6 Can be expanded into research (e.g., new algorithms).
CO7 May lead to research in deadlock prevention techniques.

PO8 with All COs

CO1 Encourages independent exploration of OS design.
CO2 Promotes self-study of algorithms.
CO3 System-level coding improves learning independence.
CO4 Requires exploring various memory strategies.
CO5 Self-learning of thread libraries and tools.
CO6 Motivates learners to understand synchronization methods.
CO7 Encourages analytical learning in deadlock scenarios.

PO9 with All COs

CO1 Builds strong digital systems understanding.
CO2 Involves modern scheduling simulators/tools.
CO3 Teaches low-level system interaction.
CO4 Core to understanding memory in tech systems.
CO5 Essential for modern parallel computing.
CO6 Integral in building safe tech environments.
CO7 Enhances skills in tech system robustness.

PO10 with All COs

CO1–CO7 These topics are technical and don't directly engage with diversity, inclusion, or empathy themes.

PO11 with All COs

CO1–CO7 OS concepts do not directly relate to environmental or ethical values. Some indirect value inculcation via responsibility in programming.

PO12 with All COs

CO1 Encourages self-learning and exploration.
CO2 Needs self-discipline in understanding algorithms.
CO3 Promotes accountable coding and debugging.
CO4 Mistakes in memory management demand accountability.
CO5 Threading errors demand responsible handling.
CO6 Requires responsible synchronization practices.
CO7 Demands careful handling of deadlocks in systems.

PO13 With all COs

CO1–CO7 These technical topics have no direct engagement with community service.

SYLLABUS (CBCS as per NEP 2024) FOR T.Y.B. Sc. (Computer Science)2024 Pattern (w. e. from AY 2026-27)

Name of the Programme : B.Sc. (Computer Science)
Program Code : **USCOS**
Class : T.Y. B.Sc.(CS)
Semester : V
Course Type : Major Mandatory (TH)
Course Name : Theoretical Computer Science
Course Code : COS-302-MRM
No. of Lectures : 30
No. of Credits : 2

Prerequisites:

- Sets, Operations on sets, Finite & infinite sets Formal Language.
- Relation, Equivalence Relation, (reflexive, transitive and symmetric closures).

Course Objectives:

- Understanding Foundations of Computation.
- To have an understanding of finite state and pushdown automata.
- To have a knowledge of regular languages and context free languages.
- To know the relation between regular language, context free language and corresponding recognizers.

Course Outcome:

CO1: Knowledge of automata, formal language theory and computability.
CO2: Demonstrate advanced knowledge of formal computation and its relationship to languages.
CO3: Distinguish different computing languages and classify their respective types.
CO4: Recognize and comprehend formal reasoning about languages.
CO5: Show a competent understanding of the basic concepts of complexity theory.
CO6: The students will be able to design.
CO7: To know basic models of information processing.

Unit No.	Topic Contents	No. of Lectures Required
Unit- I	**Finite Automata and Regular Languages** 1.1 Deterministic finite Automaton – Definition, DFA as language 1.2 Nondeterministic finite automaton 1.3 NFA TO DFA 1.4 NFA with ε- transitions 1.5 NFA with ε-Transitions to DFA & Examples 1.6 Finite automaton with output – Mealy and Moore machine	**10**
Unit- II	**Context Free Grammar and Languages** 2.1Grammar - Definition and Examples. 2.2 Derivation - Definition and Examples. 2.3 CFG: Definition & Examples. LMD, RMD, ,Parse Tree 2.4 Removing Useless Symbols 2.5 Removing unit productions 2.6 Removing ϵ productions &Nullable symbols	**08**

	2.7 Chomsky Hierarchy 2.8 Regular Grammar : Definition and types 2.9 Closure Properties of CFL's	
Unit-III	**Push Down Automaton** 3.1 Definition of PDA and examples 3.2 Construction of PDA using empty stack and final State method : Examples using stack method 3.3 Definition DPDA & NPDA, their correlation and Examples of NPDA 3.4 CFG (in GNF) to PDA : Method and examples	**06**
Unit-IV	**Turing Machine** 4.1 The Turing Machine Model and Definition of TM 4.2 Design of Turing Machines 4.3 Turing Machine Limitations 4.4 Problems on language recognizers 4.5 Language accepted by TM	**06**

References: -

1. Introduction to Automata theory, Languages and computation By John E. Hopcroft and Jeffrey Ullman – Narosa Publishing House.

2. Introduction to Automata theory, Languages and computation By John Hopcroft, Rajeev Motwani and Jeffrey Ullman –Third edition Pearson Education

3. Introduction to Computer Theory Daniel I. A. Cohen – 2nd edition – John Wiley & Sons

4. Introduction to Languages and The Theory of Computation John C. Martin TMH, Second Edition

Mapping of this course with Programme Outcomes

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	2	1	3	3	1	2	2	2	1	1	2	1
CO2	3	2	1	3	3	1	3	2	2	1	1	2	1
CO3	3	2	1	3	3	1	2	2	2	1	1	2	1
CO4	3	2	1	3	3	1	3	2	2	1	1	2	1
CO5	3	2	1	3	3	1	3	2	2	1	1	2	1
CO6	2	3	2	3	3	2	2	2	3	1	1	3	1
CO7	3	2	1	3	3	1	3	2	2	1	1	2	1

CO1: Knowledge of automata, formal language theory and computability

Strongly supports PO1, PO4, PO5 due to theoretical foundation and analytical reasoning; moderately supports practical, research, and digital skills; partially contributes to entrepreneurship and social outcomes.

CO2: Advanced knowledge of formal computation and languages

Strongly related to PO1, PO4, PO5, PO7 as it builds deep theoretical and research capability; moderate linkage to practical and digital competencies.

CO3: Distinguish and classify computing languages

Strongly supports PO1, PO4 and PO5 through conceptual clarity and classification ability; moderate contribution to applied and technical skills.

CO4: Formal reasoning about languages

Strong alignment with PO1, PO4, PO5, PO7 due to analytical reasoning and research orientation; moderate support to learning and technical skills.

CO5: Understanding of complexity theory

Strongly linked to PO1, PO4, PO5 and PO7 because of algorithmic analysis and computational complexity reasoning.

CO6: Design capability

Strongly supports PO2, PO4, PO5, PO9, PO12 due to practical implementation, technical design, and responsibility; moderate relation to collaboration and entrepreneurship.

CO7: Basic models of information processing

Strong alignment with PO1, PO4, PO5and PO7 as it strengthens theoretical foundation and research understanding.

SYLLABUS (CBCS) FOR T.Y.B.Sc. (Computer Science) (SEM-V) 2024 Pattern
(w.e.f. A.Y.-2026-2027)

Name of the Programme : B.Sc. Computer Science
Program Code : USCOS
Class : T.Y. B.Sc. (Comp. Sci.)
Semester : V
Course Type : Major (TH)
Course Name : Computer Networking
Course Code : COS-303- MRM
No. of Lectures : 30
No. of Credits : 2

Pre-requisites: Basics knowledge of computer

Objectives: This course will prepare students in Basic networking concepts.
1. Understand different types of networks, various topologies and application of networks.
2. Understand types of addresses, data communication.
3. Understand the concept of networking models, protocols, functionality of each layer.
4. Learn basic networking hardware and tools.
5. Understand wired and wireless networks, its types, functionality of layer.

Learning Outcomes:
CO1: Understanding Networking Concepts - Define and explain fundamental networking concepts, including protocols, data communication and network architectures.
CO2: Network Models - Understand and apply knowledge of OSI, TCP/IP Models.
CO3: Network Protocols – Describe and analyze various networking protocols and their Functionalities.
C04: Network Design and implementation - Design and implement LAN, WAN & MAN.
CO5: Networking devices and Technologies - Evaluate and select appropriate networking devices and technologies for specific scenarios.
CO6: Wired & Wireless Networking: Understand principles, understand design and implementation of wired & wireless communication.
CO7: Internet Technologies: Understand the functioning of the internet and related technologies.

Unit No.	Title & Contents	No. of Lectures
UNIT - I	**Introduction to Computer Network** 1.1 Computer Networks- Goals, applications 1.2 Network Hardware's – Broadcast and point to point. 1.3 Topology – Star, Bus, Mesh, Ring etc. 1.4 Network Types: LAN, MAN, WAN, Wireless Network, internetwork 1.5 Data Communication – Definition, Components, data representation, Data flow. 1.6 Protocols and Standards: Defacto, Dejure standard 1.7 Network Software- Protocol Hierarchies, Design issues of the layer, Connection and connectionless services,	06

UNIT -II	**Network Models &Transmission Media** 2.1 Reference Model – OSI Reference Model, TCP/IP Reference Model, Comparison of OSI & TCP/IP Model, 2.2 Addressing – Physical, Logical and Port addresses 2.3 Guided Media – Twisted pair cable, Coaxial Cable, Fiber optic cable 2.4 Unguided Media – Radio Waves, Micro wave Transmission, Infrared, Light wave Transmission	06
UNIT -III	**Lower layers: Physical and Data link layers** 3.1 Communication at the physical layer, Data and signals. Transmission Impairment, Data rate limits, Performance Transmission Modes. 3.2 Switching – Circuit, Message and Packet Switching. 3.3 Design issues of Data Link Layer, Services – Framing, Error control, Flow Control, Congestion Control, Link layer addressing. 3.4 Data link Protocols – simplex, stop and wait and stop and wait Automatic Repeat Request (ARQ). 3.5 Sliding Window Protocols – One-bit sliding window protocol, Pipeline technique, Go back N and Selective Repeat Automatic Repeat Request with comparison. 3.6 DLL Protocols – HDLC, PPP 3.7 Medium Access Sublayer: Random Access Protocols, Controlled Access, Channelization. 3.8 Physical and Data link layer devices – Repeater, Hubs, Bridge	10
UNIT -IV	**The Network Layer** 4.1 Design Issues, Store-and-forward packet switching, 4.2 Services Provided to the Transport Layer, Implementation of Connectionless and Connection Oriented Service, 4.3 Comparison of Virtual Circuit and Datagram subnets 4.4 LogicalAddressingIPV4Addresses–Address Space, Notations, Classful Addressing, Subnetting, Supernetting, Classless Addressing, Network Address Translation (NAT), (Problems should be covered On Addressing) 4.5 IPV4ProtocolDatagramFormat 4.6 Routing Properties of routing algorithm, Adaptive and Non-Adaptive Routing Algorithms 4.7 Congestion Control –Definition, Factors of Congestion, Difference between congestion control and flow control, General Principles and Congestion Prevention Policies 4.8 Network Layer Devices–Routers	08

Reference Books:

1) Computer Networks by Andrew Tanenbaum, Pearson Education.[4th Edition]
2) Data Communication and Networking by Behrouz Forouzan, TATA McGraw Hill.[4th/5thEd.]
3) Networking All In One Dummies Wiley Publication.[5th Edition]

CO-PO Mapping Table:

COs \ POs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	2	1	1	2	1	1	2	2	1	1	1	1
CO2	3	2	1	1	2	1	1	2	3	1	1	1	1
CO3	3	3	1	1	3	1	2	2	3	1	1	1	1
CO4	2	3	2	3	3	2	1	2	3	1	1	2	2
CO5	2	3	1	3	3	2	2	2	3	1	1	2	2
CO6	2	3	1	2	3	2	2	2	3	1	1	2	2
CO7	2	2	2	2	3	1	2	2	3	1	1	1	2

Justification for Mapping

CO1: Understanding Networking Concepts

- **PO1 (3):** Strong foundational knowledge.
- **PO2 (2):** Procedural understanding of network architecture.
- **PO5 (2):** Basic problem-solving with networking basics.
- **PO9 (2):** Digital literacy is inherent in understanding networks.
- Other POs have low or general relevance.

CO2: Network Models

- **PO1 (3):** Requires a comprehensive grasp of conceptual models.
- **PO2 (2):** Application-based procedural knowledge.
- **PO5 (2):** Analytical reasoning for model usage.
- **PO9 (3):** Crucial for understanding digital protocols.

CO3: Network Protocols

- **PO1 (3):** In-depth theoretical understanding required.
- **PO2 (3):** Strong practical and procedural applications.
- **PO5 (3):** Requires analysis and evaluation of protocol behavior.
- **PO9 (3):** Protocols are the backbone of digital communication.

CO4: Network Design and Implementation

- **PO2 (3):** Emphasizes professional/practical application.
- **PO4 (3):** Involves specialized technical skill.
- **PO5 (3):** High-level problem-solving and reasoning.
- **PO12 & PO13 (2):** Autonomy and responsibility in project work; community engagement possible in real-world implementation.

CO5: Networking Devices and Technologies

- **PO2 (3):** Selection of devices is procedural and practical.
- **PO4 (3):** Needs specialized technical knowledge.
- **PO5 (3):** Requires judgment and problem-solving.
- **PO9 (3):** Heavy emphasis on digital tools and tech.
- **PO12 & PO13 (2):** Applicable in practical network setups, especially in collaborative or community projects.

CO6: Wired & Wireless Networking

- **PO2 (3):** Practical/professional knowledge of both mediums.
- **PO4 (2):** Competency in hardware and protocols.
- **PO5 (3):** Analytical and design-based problem-solving.
- **PO9 (3):** Technologies are deeply digital.
- **PO6, PO12, PO13 (2):** Collaboration and real-world application aspects.

CO7: Internet Technologies

- **PO1 (2):** Fundamental understanding needed.
- **PO2 (2):** Functional and procedural knowledge.
- **PO3 (2):** Innovation and entrepreneurship in internet services.
- **PO5 (3):** Evaluative and design skills are key.
- **PO9 (3):** Focus on digital technologies.
- **PO13 (2):** Relevant for service and outreach initiatives.

SYLLABUS (CBCS as per NEP 2020) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern (w. e. from AY 2026-27)

Name of the Programme	**:** B.Sc. (Computer Science)
Program Code	**: USCOS**
Class	**:** T.Y. B.Sc.(CS)
Semester	**:** V
Course Type	**:** Major Mandatory (TH)
Course Name	**:** Object Oriented Software Engineering
Course Code	**:** COS-304-MRM
No. of Lectures	**:** 30
No. of Credits	**:** 2

Prerequisites: Knowledge of Classical Software Engineering

Aim: To Understand Object Oriented Modeling techniques and their applicability.

Objectives

- Understanding Object Orientation in Software engineering concepts and importance.
- Understand the Unified Modeling Language concepts, importance and its component.
- Understand Structural, Behavioral, Dynamic modeling techniques and diagrams.
- Apply Object Oriented concepts and its techniques for software development.
- Implement the Unified Modeling Language concepts, importance and its component.
- Apply Structural, Behavioral, Dynamic modeling techniques and diagrams.
- Implement the concept of Object-Oriented software development process model.

OUTCOMES:

CO1: Develop models using the UML notation.
CO2: Apply an iterative, agile process.
CO3: Analyze requirements with use cases.
CO4: Create domain models
CO5: Relate analysis and design artifacts.
CO6: Design object solutions with patterns and architectural layers.
CO7: Apply concepts to a semester-long software engineering project.

Unit No.	Title and Contents	No. of Lectures Required
Unit-I	**Object Oriented Concepts and Principles** 1.1 Introduction, Object, Classes and Instance, Polymorphism, Inheritance 1. 2 Object Oriented System Development- Introduction, Function/Data Methods (With Visibility), Object Oriented Analysis, Object Oriented Construction 1.2 Identifying the Elements of an Object Model Aggregations, 1.3 Identifying Classes and Objects, Identity, Dynamic binding, Persistence, Meta classes 1.5 Specifying the Attributes (With Visibility) 1.6 Defining Operations 1.7 Finalizing the Object Definition	**04**
Unit-II	**Introduction to UML and Object Oriented Methodology** 2.1 Concept of UML 2.2 Advantages of UML 2.3 Object oriented Methods (The Booch Method, The Coad &Yourdon Method, Jacobson Method and Raumbaugh Method)	**02**
Unit-III	**Structural Modeling** 3.1 Classes 3.2 Relationship (Interface, Types and Roles, Packages , Common Mechanism) 3.4 Class Diagram (Minimum three examples should be covered) 3.5 Object Diagram (Minimum three examples should be covered) **Behavioral Modeling** 3.6 Interactions 3.7 Use Cases and Use Case Diagram with stereo types (Minimum three examples should be covered) 3.8 Interaction Diagram (Minimum two examples should be covered) 3.9 Sequence Diagram (Minimum two examples should be covered) 3.10 Activity Diagram (Minimum two examples should be covered) 3.11 State Chart Diagram (Minimum two examples should be covered)	**14**
Unit-IV	**Object Oriented Analysis, Architectural modeling, Testing** 4.1 Iterative Development and the Rational Unified Process 4.2 Inception 4.3 Understanding Requirements 4.4 Use Case Model from Inception to Elaboration 4.5 Elaboration 4.6 Component 4.7 Components Diagram (Minimum two examples should be covered) 4.8 Deployment Diagram (Minimum two examples should be covered) 4.9 Collaboration Diagram (Minimum two examples should be covered) Object Oriented Testing Strategies 4.10 Test Case Design for Object Oriented Software 4.11 Inter Class Test Case Design(Use of any freeware designing tool)	**10**

References

1. Ivar Jacobson, "Object Oriented Software Engineering", Pearson Education INC
2. Craig Larman, "Applying UML and Patterns" Pearson Education INC
3. Bennett, Simon, "Object Oriented Analysis and Design" McGraw Hill
4. Ali Bahrami, "Object Oriented System Development", McGraw Hill International Edition, 2008
5. Brahma Dathan, Sarnath Ramnath, "Object-Oriented Analysis, Design and Implementation", Universities Press, 2010
6. Bernd Bruegge, Allen H. Dutoit, Object Oriented Software Engineering using UML,Patterns and Java, Pearson 2004
7. Craig Larman, Applying UML and Patterns – An Introduction to Object-Oriented Analysis and Design and Iterative Development" , 3rd Edition, Pearson Education, 2005
8. Grady Booch, James Rumbaugh, Ivar Jacobson, "The Unified Modeling Language User Guide", Addison Wesley Long man, 1999
9. Martin Fowler, "UML Distilled A Brief Guide to Standard Object Modeling Language", 3rd Edition, Addison Wesley, 2003
10. Russ Miles, Kim Hamilton, "Learning UML 2.0", O'Reilly, 2008

Mapping of this course with Programme Outcomes

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	3	1	3	3	2	2	2	3	1	1	2	1
CO2	3	3	2	3	3	3	2	3	3	2	2	3	2
CO3	3	3	2	3	3	3	2	2	3	2	1	2	1
CO4	3	3	2	3	3	2	2	2	3	1	1	2	1
CO5	3	3	2	3	3	2	2	2	3	1	1	2	1
CO6	3	3	2	3	3	2	2	3	3	2	2	3	2
CO7	3	3	3	3	3	3	2	3	3	2	2	3	3

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification

PO1: All COs strongly build theoretical and conceptual understanding of software engineering principles and modeling techniques.

PO2: All COs involve hands-on modeling, design, and project execution reflecting strong professional application.

PO3: Agile application and semester project (CO7) strongly support entrepreneurial thinking; modeling/design outcomes moderately support it.

PO4: UML, domain modeling, patterns, and architecture (CO1–CO6) directly develop specialized software engineering competencies.

PO5: Requirements analysis, modeling, and design decisions across all COs require strong analytical and problem-solving skills.

PO6: Agile process, use cases, and semester project strongly enhance teamwork and communication; modeling tasks moderately support it.

PO7: Requirement analysis, domain modeling, and project work promote research, investigation, and design evaluation skills.

PO8: Iterative learning through agile process and project work fosters continuous learning capability.

PO9: All COs require use of software tools, modeling platforms, and development environments, strongly supporting digital competence.

PO10: Agile teamwork and project collaboration moderately promote inclusivity and empathy in team-based environments.

PO11: Ethical design, sustainable software thinking, and responsible project execution moderately relate to values and awareness.

PO12: Independent modeling, design responsibility, and project accountability strongly relate to autonomy and ownership.

PO13: Semester-long project (CO7) strongly supports community engagement; other outcomes partially relate through collaborative practices.

SYLLABUS (CBCS-2024 Pattern as per NEP 2020) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern (w. e. from June 2026)

Name of the Programme	**:** B.Sc. (Computer Science)
Program Code	**: USCOS**
Class	**:** T.Y. B.Sc.(CS)
Semester	**:** V
Course Type	**:** Major Mandatory (PR)
Course Name	**:** Lab Course on Advanced Java
Course Code	**:** COS-305-MRM
No. of Practical	**:** 15
No. of Credits	**:** 2

PREREQUISITES:

- Basic knowledge of Core Java programming concepts (OOP, class, object, inheritance, polymorphism).
- Understanding of exception handling and file handling in Java.
- Familiarity with basic data structures.
- Basic understanding of relational database concepts (tables, keys, SQL queries).
- Fundamental knowledge of HTML.

COURSE OBJECTIVES:

- To understand and implement Java Collection Framework for efficient data handling.
- To develop multithreaded applications using thread lifecycle and synchronization mechanisms.
- To design user-friendly graphical user interfaces using Swing components and event handling.
- To establish database connectivity using JDBC and perform CRUD operations.
- To develop dynamic web applications using JSP.
- To integrate GUI, database, and web technologies for real-world application development.
- To enhance problem-solving and application development skills using advanced Java technologies.

COURSE OUTCOMEs:

CO1: Apply appropriate Collection classes for efficient data storage and manipulation.
CO2: Design and implement multithreaded programs using synchronization.
CO3: Develop event-driven GUI applications using Swing framework.
CO4: Connect Java applications to databases and perform secure data operations.
CO5: Create dynamic web pages using JSP and handle client-server interaction.
CO6: Develop integrated applications combining GUI, database, and web modules.
CO7: Deploy and test Java-based applications following standard programming practices.

Assignment No.	Name of Assignment	No. of Practicals Required
1	Collection	2
2	Thread	2
3	GUI Designing using Swing	4
4	Database	2
5	JSP	2
6	Mini Project	3

Mapping PO's With CO's (3: Strongly Related, 2: Moderately Related, 1: Partially Related)

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	3	1	3	3	1	1	2	3	1	1	2	1
CO2	3	3	1	3	3	1	1	2	3	1	1	2	1
CO3	2	3	1	3	3	1	1	2	3	1	1	3	1
CO4	2	3	1	3	3	1	1	2	3	1	1	3	1
CO5	3	3	1	3	3	1	1	2	3	1	1	2	1
CO6	2	3	1	3	3	1	1	2	3	1	1	3	1
CO7	3	2	1	2	3	1	2	3	2	1	1	1	1

PO1: Comprehensive Knowledge and Understanding
CO1 – CO7: The course covers advanced Java concepts including Collections, Multithreading, Swing GUI, JDBC, and JSP technologies. Students understand programming paradigms, architecture, and implementation techniques used in real-world applications. This builds strong conceptual knowledge of Java application development and computing principles.

PO2: Practical, Professional, and Procedural Knowledge
CO1 – CO7: Students perform hands-on laboratory exercises such as database connectivity, GUI designing, session handling, and web programming. The course directly develops professional programming practices, debugging, and application deployment skills required in software development industries.

PO3: Entrepreneurial Mindset and Knowledge
CO5, CO6: Development of web-based applications and integrated systems enables students to understand how software products are created and delivered. This supports innovation and small-scale software solution development though entrepreneurship is not the primary focus.

PO4: Specialized Skills and Competencies
CO1 – CO7: Students gain specialized technical competencies in Java technologies such as Swing, JDBC, and JSP. They implement event-driven programming, concurrent programming, and database-driven applications demonstrating professional-level software development capability.

PO5: Capacity for Application, Problem-Solving, and Analytical Reasoning
CO1, CO2, CO4, CO6: Students analyze programming problems, choose appropriate data structures, manage thread synchronization, and integrate multiple modules. They develop logical thinking and debugging skills required for solving real-world computing problems.

PO6: Communication Skills and Collaboration
CO3, CO5, CO7: Students design user interfaces, document programs, and demonstrate working applications during practical sessions and mini-projects. This improves technical communication and teamwork during project implementation.

PO7: Research-related Skills
CO2, CO4, CO6: Students explore alternative programming approaches, experiment with different database queries, and test performance of concurrent applications. They learn systematic experimentation and result verification.

PO8: Learning How to Learn Skills
CO1 – CO7: Students learn new APIs, libraries, and frameworks independently while completing practical assignments and mini projects. The course encourages self-learning through debugging, documentation reading, and experimentation.

PO9: Digital and Technological Skills
CO1 – CO7: Students use IDEs, application servers, database servers, and programming frameworks. They develop full-stack Java applications demonstrating strong technological proficiency.

PO10: Multicultural Competence, Inclusive Spirit, and Empathy
CO3, CO5: GUI and web application development considers usability and user-centric design principles for diverse users, though indirectly related to multicultural competence.

PO11: Value Inculcation and Environmental Awareness
CO7: Students follow ethical coding practices, proper software usage, and responsible computing practices during development and deployment.

PO12: Autonomy, Responsibility, and Accountability
CO6, CO7: Students independently develop complete applications and are responsible for testing and debugging programs during practical evaluation and mini-projects.

PO13: Community Engagement and Service
CO5, CO6: The course enables development of applications such as management systems, service portals, and utility software which can be used for institutional or community purposes.

SYLLABUS (CBCS as per NEP 2023) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern (w. e. from AY 2025-26)

Name of the Programme : B.Sc. Computer Science
Program Code : USCOS
Class : T.Y. B.Sc. (Comp. Sci.)
Semester : V
Course Type : Major Mandatory (PR)
Course Name : Lab Course based on COS-301-MRM
Course Code : COS-306-MRM
No. of Practical : 15
No. of Credits : 2
Prerequisites:
• Basic Knowledge of Computer

Course Objectives:

- To understand Complexity of Operating system as a software. .
- To understand design issues related to process management and various related algorithms
- To understand design issues related to Deadlock
- To understand various commands executed in different OS
- To understand various basic Networking Commands.

Course Outcome:

CO1: Working of Shell and system call
CO2: Implementation of process scheduling
CO3: Implementation of deadlock avoidance algorithm
CO4: Study the role of the operating system in processes.
CO5: Learn the Multithreading Concepts
CO6: Learn networking essentials
CO7: Learn network administration and configuration

Assignment No.	Name of Assignment	No. of Practical Required
1.	Linux Commands	01
2.	UNIX Shell Programming	02
3.	CPU Scheduling	04
4.	Bankers Algorithm	02
5.	Thread Scheduling	02
6.	Case Study	01
7.	Transmission Media Assignment	01
8.	Network Configuration	01
9.	Network devices	01

Mapping of PO and CO with Justification

((3: Strongly Related, 2: Moderately Related, 1: Partially Related)

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	3	2	3	3	2	2	2	3	1	1	2	1
CO2	3	3	2	3	3	2	1	2	3	1	1	2	1
CO3	3	3	2	3	3	2	1	2	3	1	1	2	1
CO4	3	3	2	3	3	1	1	2	3	1	1	2	1
CO5	2	3	3	3	3	2	1	2	3	1	1	2	1
CO6	2	3	3	3	3	3	2	2	3	2	2	3	2
CO7	2	3	3	3	3	3	2	2	3	2	2	3	2

Justification:

PO1 with All COs

CO1: Builds strong conceptual foundation about shell and system calls, which are core components of OS.

CO2: In-depth knowledge of how CPU scheduling works improves understanding of OS internals.

CO3: Studying deadlock avoidance deepens theoretical knowledge and resource allocation strategies.

CO4: Covers essential concepts of memory hierarchy and management.

CO5: Offers insight into concurrency models and OS-level multithreading.

CO6: Introduces networking concepts essential for modern OS design.

CO7: Helps understand administrative tasks and real-world network configuration as part of OS functionality.

PO2 with All COs

CO1: Practical understanding of how shell and system calls operate within an OS.

CO2: Hands-on implementation of scheduling algorithms demonstrates real-time process control.

CO3: Deadlock handling simulations develop procedural problem-solving skills.

CO4: Demonstrates memory allocation methods in actual systems.

CO5: Applying multithreading principles enhances procedural concurrency skills.

CO6: Involves practical labs on network essentials with OS interaction.

CO7: Involves configuring and troubleshooting networks—a professional skill in admin roles.

PO3 with All COs

CO1: Encourages creative system-level thinking in shell programming.

CO2: Process scheduling knowledge can inspire optimization tools or products.

CO3: Encourages thinking about innovative solutions for system deadlocks.

CO4: Sparks entrepreneurial thinking for building efficient memory solutions.

CO5: Multithreading concepts support ideas in developing performance-oriented applications.

CO6: Enables understanding of network essentials for tech-based startups.

CO7: Useful for entrepreneurial efforts in system administration or network solutions.

PO4 with All COs

CO1: Skill in writing and debugging system-level programs.

CO2: Specialization in designing and analyzing scheduling systems.

CO3: Expertise in handling resource allocation and system stability.
CO4: Deepens knowledge of memory layout, paging, segmentation.
CO5: Prepares for advanced roles in multi-core and parallel programming.
CO6: Develops essential networking skills critical for specialized OS tasks.
CO7: Prepares students for system/network admin roles with configuration expertise.

PO5 with All COs

CO1: Requires analytical thinking to understand shell behavior and system services.
CO2: Problem-solving in implementing and analyzing scheduling algorithms.
CO3: Encourages developing strategies to avoid deadlock scenarios.
CO4: Requires reasoning about memory utilization and management techniques.
CO5: Involves solving concurrency problems using multithreading.
CO6: Requires analytical setup and troubleshooting of networks.
CO7: Involves applying knowledge to solve configuration and network issues.

PO6 with All COs

CO1: Basic system programming discussions help improve technical communication.
CO2: Collaborative coding in scheduling projects.
CO3: Peer reviews on deadlock logic help develop clarity in explanation.
CO4: Less emphasis here, but discussions improve teamwork.
CO5: Working in teams to simulate thread synchronization.
CO6: Strong teamwork and communication needed during networking labs.
CO7: Real-world admin tasks often require group coordination.

PO7 with All COs

CO1: Introduces basic investigation of how systems operate internally.
CO2: Can inspire scheduling optimization research.
CO3: Encourages studying resource-allocation models.
CO4: Triggers investigation into memory management research topics.
CO5: Can lead to exploring performance benchmarks for threading models.
CO6: Opens inquiry into protocol design and network performance.
CO7: Research-driven understanding of secure and optimized configurations.

PO8 with All COs

CO1–CO7: Each CO contributes by requiring students to independently explore and learn system internals, configurations, and concepts through hands-on labs, manuals, and simulation tools.

PO9 with All COs

CO1–CO7: All outcomes heavily support this PO. They involve command-line tools, system programming, OS configuration, threading, and networking—core digital competencies.

PO10 with All COs

CO1–CO7: Indirectly supported when students collaborate across cultures during group tasks or peer reviews. Networking and OS administration might also include diverse user considerations.

PO11 with All COs

CO1–CO7: While technical, responsible resource usage (like CPU, memory, network bandwidth) can help foster eco-conscious software design.

PO12 with All COs

CO1–CO7: All tasks promote independent coding, debugging, and problem-solving. Configuration and administration tasks develop a sense of responsibility and accountability.

PO13 with All COs

CO1–CO7: Especially CO6 and CO7 foster skills that can be applied in community tech support, digital literacy training, or volunteer IT setup in local networks.

SYLLABUS (CBCS as per NEP 2020) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern (w. e. from AY 2026-27)

Name of the Programme : B.Sc. (Computer Science)
Program Code : **USCOS**
Class : T.Y. B.Sc. (C.S.)
Semester : V
Course Type : Elective (TH)
Course Name : **Core PHP**
Course Code : COS-307-MJE(A)
No. of Lectures : 30
No. of Credits : 2

Prerequisites:
• HTML,CSS
Objectives:

- To learn basics of PHP.
- To design dynamic, interactive web pages.
- To learn string and arrays in PHP.
- To learn about sanitizing user inputs.
- To learn object oriented programming.
- To learn database connectivity with PHP.
- To understand how to structure code efficiently.

Outcome :

CO1.Grasp the fundamentals of PHP syntax and programming constructs.
CO2. Develop dynamic and interactive web pages using PHP.
CO3. Understand the various string related operations in PHP.
CO4.Understand basic security practices, such as sanitizing user inputs and preventing SQL injection.
CO5. Gain basic knowledge of PHP's object-oriented features like classes, objects, and inheritance.
CO6. Gain an understanding of how PHP interacts with web servers and databases.
CO7. Design and structure code efficiently, promoting readability and reusability.

UNIT No.	Chapter Name with Topics	No. of Lectures Required
UNIT-I	**Introduction to PHP** 1.1 Lexical structure 1.2 Language basics 1.3 Defining and calling a function 1.4 Default parameters 1.5 Variable parameters, Missing parameters 1.6 Variable function, Anonymous function	**04**
UNIT- II	**String and Arrays** 2.1 Types of strings in PHP	**08**

	2.2 Comparing strings 2.3 Manipulating and searching strings 2.4 Regular Expressions 2.5 Indexed Vs Associative arrays 2.6 Storing data in arrays 2.7 Multidimensional arrays 2.8 Extracting multiple values 2.9 Sorting 2.10 Action on entire arrays	
UNIT-III	**Introduction to Object Oriented Programming** 3.1 Classes and Objects 3.2 Inheritance 3.3 Interfaces 3.4 Encapsulation 3.5 Traits 3.6 Autoloading classes 3.7 Exception handling 3.8 Predefined exceptions 3.9 Namespaces in OOP in PHP 3.10 Predefined PHP classes and interfaces	**12**
UNIT-IV	**Databases (PHP-PostgreSQL)** 4.1 Introduction to PDO 4.2 Predefined constants 4.3 Supported databases 4.4 The PDO class 4.5 PDO class methods 4.6 Security using PDO 4.7 PDOStatement class 4.8 Create, Read, Update and Delete (CRUD) operations	**06**

Mapping of this course with Programme Outcomes

Course Outcomes	Programme Outcomes(POs)												
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	2	1	2	2	1	1	2	2	1	1	1	1
CO2	3	3	2	3	3	2	2	3	3	2	2	2	2
CO3	3	2	1	2	2	1	1	2	2	1	1	1	1
CO4	2	3	2	3	3	1	2	3	3	2	2	3	2
CO5	3	3	2	3	3	2	2	2	3	1	1	2	2
CO6	3	3	2	3	3	2	2	3	3	2	2	2	2
CO7	2	3	2	3	3	2	2	3	3	2	2	3	2

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification of mapping

Mapping of PO1 to all COs

CO1:PO1- Strongly relates as understanding syntax is fundamental knowledge.

CO2:PO1- Applies fundamental knowledge in real-world use.

CO3:PO1- Strongly tied to core programming understanding.

CO4:PO1- Requires solid theoretical understanding of vulnerabilities.

CO5:PO1- Deepens understanding of advanced programming concepts.

CO6:PO1- Strongly enhances understanding of web backend logic.
CO7:PO1- Relates to better understanding of coding principles.

Mapping of PO2 to all COs

CO1:PO2- Moderately related since basic syntax supports practical knowledge.
CO2:PO2- Strongly involves procedural and practical knowledge.
CO3:PO2- Moderately relates to practical usage of strings.
CO4:PO2- Strongly tied to professional and procedural application.
CO5:PO2- Strongly relates to procedural and professional design.
CO6:PO2- Core procedural skill for web programming.
CO7:PO2- Strongly tied to professional coding practices.

Mapping of PO3 to all COs

CO1:PO3- Partially helps in initiating simple entrepreneurial projects.
CO2:PO3- Enables creating business websites/applications.
CO3:PO3- Partial use in formatting content for business tools.
CO4:PO3- Important for secure business application development.
CO5:PO3- Useful in scalable entrepreneurial projects.
CO6:PO3- Essential for database-backed business apps.
CO7:PO3- Supports maintainable, scalable business solutions.

Mapping of PO4 to all COs

CO1:PO4- Provides foundational programming skills.
CO2:PO4- Strongly relates to developing specialized coding skills.
CO3:PO4- Enhances problem-solving through string manipulations.
CO4:PO4- Vital skill in secure coding.
CO5:PO4- Builds competency in software development.
CO6:PO4- Specialized web development competency.
CO7:PO4- Key skill for software competency.

Mapping of PO5 to all COs

CO1:PO5- Involves logical thinking and problem-solving basics.
CO2:PO5- Involves problem-solving for dynamic page generation.
CO3:PO5- Involves logic in handling and parsing string data.
CO4:PO5- Involves analytical reasoning for security loopholes.
CO5:PO5- Promotes structured thinking and reuse.
CO6:PO5- Involves logic and application in DB handling.
CO7:PO5- Strongly linked to problem-solving via clean code.

Mapping of PO6 to all COs

CO1:PO6- Indirectly aids in communicating logic through code.
CO2:PO6- Supports team collaboration in developing web projects.
CO3:PO6- Aids communication in terms of user-facing content.
CO4:PO6- Indirect role in team discussions on security practices.
CO5:PO6- Enhances teamwork through modular code design.
CO6:PO6- Collaboration in web and DB integration projects.
CO7:PO6- Improves teamwork and code sharing.

Mapping of PO7 to all COs

CO1:PO7- Minor support to research via foundational knowledge.

CO2:PO7- Helps design research tools like data collection sites.
CO3:PO7- Minor role in data formatting for research tools.
CO4:PO7- Research relevance in secure data collection.
CO5:PO7- Supports research tool modularization.
CO6:PO7- Useful for research involving backend data storage.
CO7:PO7- Structured code helps in research simulations.

Mapping of PO8 to all COs

CO1:PO8- Learning PHP syntax supports independent learning.
CO2:PO8- Strongly aids in learning by doing approach.
CO3:PO8- Helps build learning patterns through examples.
CO4:PO8- Encourages continuous learning due to evolving threats.
CO5:PO8- Promotes continued learning through abstraction.
CO6:PO8- Promotes hands-on learning.
CO7:PO8- Encourages reflective learning and improvement.

Mapping of PO9 to all COs

CO1:PO9- Enhances basic digital literacy through scripting.
CO2:PO9- Enhances use of digital and tech tools.
CO3:PO9- Relates to technical processing and output.
CO4:PO9- Strongly linked to secure tech implementation.
CO5:PO9- Critical for modern object-oriented development.
CO6:PO9- Deepens digital and tech capabilities.
CO7:PO9- Deep tech skill in design patterns and clarity.

Mapping of PO10 to all COs

CO1:PO10- Minimal impact; may help in inclusive tech practices.
CO2:PO10- Can involve inclusive web development practices.
CO3:PO10- Limited role, may aid in content personalization.
CO4:PO10- Ensures inclusive and ethical handling of user data.
CO5:PO10- Minimal direct impact.
CO6:PO10- Inclusion via accessible, data-driven design.
CO7:PO10- Supports inclusive coding through documentation.

Mapping of PO11 to all COs

CO1:PO11- Can support digital solutions for environmental awareness.
CO2:PO11- Web solutions can target environmental causes.
CO3:PO11- Could support formatting eco-focused messages.
CO4:PO11- Promotes values in ethical programming.
CO5:PO11- Object-oriented methods can apply in environmental software.
CO6:PO11- Useful in environmental data tracking.
CO7:PO11- Can be used in reusable environmental tools.

Mapping of PO12 to all COs

CO1:PO12- Slightly related to individual coding responsibility.
CO2:PO12- Requires accountability in code development.
CO3:PO12- Encourages responsibility in data handling.
CO4:PO12- Reinforces accountability in code responsibility.
CO5:PO12- Encourages ownership in design decisions.

CO6:PO12- Requires accountability in DB interactions.
CO7:PO12- High level of responsibility in code clarity.

Mapping of PO13 to all COs

CO1:PO13- Can indirectly support web solutions for community needs.
CO2:PO13- Directly applicable to service-oriented web solutions.
CO3:PO13- Can support basic string processing in web forms.
CO4:PO13- Security is critical in public service platforms.
CO5:PO13- Supports scalable and reusable service applications.
CO6:PO13- Applicable in public service platforms.
CO7:PO13- Reusable code useful for community apps.

SYLLABUS (CBCS as per NEP 2020) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern

(w. e. from AY 2026-27)

Name of the Programme : B.Sc. (Computer Science)
Program Code : **USCOS**
Class : T.Y. B.Sc. (C.S.)
Semester : V
Course Type : Elective (TH)
Course Name : **Blockchain Technology**
Course Code : COS-307-MJE(B)
No. of Lectures : 30
No. of Credits : 2

On completion of the course, students will be able to:

1. **CO1:** Understand the foundational concepts, history, and key components of Blockchain Technology.
2. **CO2:** Analyze cryptographic algorithms (Hashing, Digital Signatures) used to secure blockchain.
3. **CO3:** Evaluate different consensus mechanisms (PoW, PoS) and their trade-offs.
4. **CO4:** Understand the architecture of Bitcoin and Ethereum, including account structures and transactions.
5. **CO5:** Design and develop decentralized applications (DApps) using Smart Contracts.
6. **CO6:** Explore Hyperledger Fabric for enterprise blockchain applications.
7. **CO7:** Analyze security risks, challenges, and future trends in blockchain technology.

Unit No.	Chapter Name with Topics	No. of Lectures Required
UNIT-I	**Introduction to Blockchain** 1.1 Introduction to Distributed Ledger Technology (DLT). 1.2 History: Bitcoin and Satoshi Nakamoto. 1.3 Blockchain Structure: Blocks, Headers, Hashes, Merkle Trees. 1.4 Public, Private, and Consortium Blockchains.	**08**
UNIT-II	**Cryptography & Consensus Mechanisms** 2.1 Cryptographic Primitives: Hashing ($SHA{-}256$), Digital Signatures (ECDSA). 2.2 Consensus Mechanisms: Proof of Work (PoW), Proof of Stake (PoS), Delegated PoS (DPoS), Practical Byzantine Fault Tolerance (PBFT). 2.3 Mining and Validation Processes.	**08**
UNIT-III	**Ethereum and Smart Contracts** 3.1 Ethereum Virtual Machine (EVM) and Account Models. 3.2 Smart Contracts: Definition, Lifecycle, and Deployment.	**08**

	3.3 Introduction to Solidity Programming language. 3.4 Decentralized Applications (DApps).	
UNIT-IV	**Advanced Topics and Applications** 4.1 Enterprise Blockchain: Hyperledger Fabric Architecture. 4.2 Use Cases: Finance, Supply Chain, Healthcare, Voting. 4.3 Blockchain Security: Vulnerabilities, Privacy, and Scalability Issues.	**06**

Recommended References

1. Blockchain: Blueprint for a New Economy by Melanie Swan.
2. Mastering Bitcoin by Andreas M. Antonopoulos.
3. Mastering Ethereum by Andreas M. Antonopoulos, Gavin Wood.

Mapping of this course with Programme Outcomes

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	2	2	2	2	1	2	2	3	1	1	2	1
CO2	3	3	1	3	3	1	3	2	3	1	1	2	1
CO3	3	3	2	3	3	2	3	3	3	2	3	2	2
CO4	3	2	3	3	2	1	2	2	3	1	1	2	1
CO5	3	2	3	3	3	2	2	3	3	2	1	3	2
CO6	3	2	3	3	3	3	3	3	3	2	2	3	2
CO7	3	3	3	3	3	3	3	3	3	2	2	3	2

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification

PO1: All COs collectively build strong theoretical and conceptual understanding of blockchain foundations, architecture, security, and applications.

PO2: Cryptography (CO2), consensus (CO3), DApp development (CO5), and Hyperledger (CO6) strongly reflect professional and procedural expertise.

PO3: Smart contracts and enterprise blockchain (CO5, CO6) strongly support innovation and entrepreneurial applications; foundational concepts moderately support it.

PO4: Cryptographic algorithms, consensus design, smart contract development, and enterprise frameworks develop specialized blockchain competencies.

PO5: Security analysis, consensus trade-offs, and DApp development require strong analytical and problem-solving skills.

PO6: Enterprise blockchain projects (CO6) and security analysis (CO7) moderately support collaboration and technical communication.

PO7: Evaluation of consensus mechanisms and security risks strongly promotes research orientation and investigation skills.

PO8: Exploring evolving technologies like Ethereum, Hyperledger, and future trends promotes continuous and self-directed learning.

PO9: All COs require hands-on engagement with digital platforms, blockchain frameworks, and cryptographic tools.

PO10: Blockchain's decentralized global nature and enterprise applications moderately relate to multicultural and inclusive technology practices.

PO11: Consensus energy impact (PoW), ethical implications, and security awareness strongly connect with environmental and value-based considerations.

PO12: Developing DApps and enterprise blockchain solutions requires responsibility, independent work, and accountability.

PO13: Blockchain applications for public systems and decentralized services moderately contribute to community-oriented technological solutions.

SYLLABUS (CBCS as per NEP 2024) FOR T.Y.B. Sc. (Computer Science) 2024 Pattern (w. e. from AY 2026-27)

Name of the Programme : B.Sc. (Computer Science)
Program Code : **USCOS**
Class : T.Y. B.Sc.(CS)
Semester : V
Course Type : Elective (PR)
Course Name : Lab Course based on COS-307-MJE(A)
Course Code : COS-308-MJE(A)
No. of practical : 15
No. of Credits : 2
Prerequisite: HTML

Objectives:

- To design dynamic, interactive web pages.
- To learn the server side scripting language.
- To learn database connectivity with PHP

Outcome:

CO1. Learn the environment of Server Side Scripting.

CO2. Learn the use of control structures and numerous native data types

CO3. Design web pages with the ability to retrieve and present data from a database.

CO4. Learn the basic building blocks of PHP like strings, functions, arrays, objects.

CO5.Compare and contrast between Client Side Script & Server Side Script.

CO6.Build PHP script to create dynamic web content.

CO7. Compare between PHP and other server side scripting languages.

Assignment No.	Name of Assignment	No. of Practical Required
1.	Basic programs using control structures	1
2.	Functions	2
3.	Strings	3
4.	Arrays	3
5.	Object Oriented Programming	3
6.	Databases	3

CO–PO Mapping

CO \ PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	3	2	1	2	2	1	1	2	3	1	1	2	1
CO2	3	3	1	3	3	1	1	2	3	1	1	2	1
CO3	3	3	2	3	3	2	2	2	3	1	1	2	1
CO4	3	3	1	3	2	1	1	2	3	1	1	2	1
CO5	2	2	2	2	2	2	1	2	2	1	1	2	1
CO6	3	3	3	3	3	2	2	2	3	1	1	3	2
CO7	2	2	3	2	2	2	1	2	2	1	1	2	1

Justification for Each CO–PO Mapping

CO1: Learn the environment of Server-Side Scripting

- **PO1 (3):** Builds foundational knowledge of server-side architecture and execution models.
- **PO2 (2):** Provides procedural understanding of server setup and configuration.
- **PO3 (1):** Limited exposure to business implications of server-side technologies.
- **PO4 (2):** Develops technical familiarity with scripting environments.
- **PO5 (2):** Enhances understanding of backend logic flow and execution.
- **PO6 (1):** Minimal communication involvement at this stage.
- **PO7 (1):** Limited direct research orientation.
- **PO8 (2):** Encourages independent exploration of server environments.
- **PO9 (3):** Strongly develops digital and technological competence.
- **PO10 (1):** Minimal multicultural exposure.
- **PO11 (1):** Limited environmental or value-based linkage.
- **PO12 (2):** Encourages responsible configuration and deployment practices.
- **PO13 (1):** Minimal community engagement component.

CO2: Use of Control Structures and Native Data Types

- **PO1 (3):** Strengthens core programming knowledge.
- **PO2 (3):** Develops procedural programming competence.
- **PO3 (1):** Limited entrepreneurial linkage.
- **PO4 (3):** Builds strong coding and technical skills.
- **PO5 (3):** Enhances logical thinking and analytical reasoning.
- **PO6 (1):** Minimal communication skill development.
- **PO7 (1):** Limited research-related exposure.
- **PO8 (2):** Promotes self-learning of language constructs.
- **PO9 (3):** Strong digital programming skill development.
- **PO10 (1):** No direct multicultural relevance.
- **PO11 (1):** No environmental linkage.
- **PO12 (2):** Encourages responsible coding practices.
- **PO13 (1):** Limited community involvement.

CO3: Design Web Pages with Database Integration

- **PO1 (3):** Integrates frontend and backend knowledge.
- **PO2 (3):** Develops professional web development skills.
- **PO3 (2):** Supports entrepreneurial web application development.
- **PO4 (3):** Enhances database and scripting competencies.
- **PO5 (3):** Requires strong problem-solving and debugging ability.
- **PO6 (2):** Supports collaborative web development projects.
- **PO7 (2):** Encourages exploratory database-driven solutions.
- **PO8 (2):** Develops self-directed integration skills.
- **PO9 (3):** Strongly builds technological and database skills.

- **PO10 (1):** Limited cultural dimension.
- **PO11 (1):** Minimal environmental link.
- **PO12 (2):** Encourages responsible handling of user data.
- **PO13 (1):** Limited service-learning connection.

CO4: Basic Building Blocks of PHP

- **PO1 (3):** Establishes comprehensive programming foundation.
- **PO2 (3):** Develops professional scripting competence.
- **PO3 (1):** Limited entrepreneurial exposure.
- **PO4 (3):** Strong technical programming skills.
- **PO5 (2):** Supports structured coding and logic formation.
- **PO6 (1):** Minimal communication linkage.
- **PO7 (1):** Limited research relevance.
- **PO8 (2):** Encourages independent practice.
- **PO9 (3):** Strong digital skill development.
- **PO10 (1):** No direct inclusive relevance.
- **PO11 (1):** Minimal value-based linkage.
- **PO12 (2):** Encourages responsible coding standards.
- **PO13 (1):** Limited community orientation.

CO5: Compare Client-Side & Server-Side Scripts

- **PO1 (2):** Enhances conceptual understanding of web architectures.
- **PO2 (2):** Builds comparative procedural knowledge.
- **PO3 (2):** Helps evaluate business-driven technology choices.
- **PO4 (2):** Develops understanding of different implementation skills.
- **PO5 (2):** Encourages analytical comparison skills.
- **PO6 (2):** Improves ability to explain technical differences.
- **PO7 (1):** Limited research focus.
- **PO8 (2):** Encourages continuous learning of new tools.
- **PO9 (2):** Develops broader digital platform understanding.
- **PO10 (1):** Minimal multicultural exposure.
- **PO11 (1):** No environmental connection.
- **PO12 (2):** Encourages responsible technology selection.
- **PO13 (1):** Limited service impact.

CO6: Build PHP Scripts for Dynamic Content

- **PO1 (3):** Demonstrates comprehensive backend knowledge.
- **PO2 (3):** Develops professional development capability.
- **PO3 (3):** Enables creation of startup-ready dynamic web solutions.
- **PO4 (3):** Strong coding and deployment competencies.
- **PO5 (3):** Requires high analytical and debugging skills.
- **PO6 (2):** Supports collaborative software development.
- **PO7 (2):** Encourages experimentation and refinement.
- **PO8 (2):** Promotes independent project building.
- **PO9 (3):** Strong digital and web technology skills.
- **PO10 (1):** Limited inclusive aspect.
- **PO11 (1):** Minimal environmental linkage.
- **PO12 (3):** Develops accountability in handling dynamic data.
- **PO13 (2):** Supports development of community-oriented web services.

CO7: Compare PHP with Other Server-Side Languages

- **PO1 (2):** Broadens conceptual understanding of backend technologies.
- **PO2 (2):** Enhances professional tool comparison skills.
- **PO3 (3):** Supports informed entrepreneurial technology selection.

- **PO4 (2):** Develops cross-technology competency awareness.
- **PO5 (2):** Encourages critical evaluation of frameworks.
- **PO6 (2):** Improves technical communication and justification skills.
- **PO7 (1):** Limited research linkage.
- **PO8 (2):** Promotes continuous technology learning.
- **PO9 (2):** Enhances digital ecosystem awareness.
- **PO10 (1):** Minimal inclusive exposure.
- **PO11 (1):** Limited environmental connection.
- **PO12 (2):** Encourages responsible technology decision-making.
- **PO13 (1):** Limited direct community engagement.

SYLLABUS (CBCS as per NEP 2020) FOR T.Y.B. Sc. (Computer Science)

(w. e. from AY 2026-27)

Name of the Programme : B.Sc. (Computer Science)
Program Code : **USCOS**
Class : T.Y. B.Sc. (C.S.)
Semester : V
Course Type : Elective (PR)
Course Name : Lab Course based on COS-307-MJE(B)
Course Code : COS-308-MJE(B)
No. of practical : 15
No. of Credits : 2

Learning Outcomes:

CO1: Implement cryptographic hashing to secure data integrity.

CO2: Develop basic blockchain structures and consensus mechanisms.

CO3: Write, compile, and deploy smart contracts using Solidity.

CO4: Understand Ethereum network concepts (MetaMask, Testnets).

CO5: Design and interact with Decentralized Applications (dApps).

CO6: Analyze blockchain transaction flows and account structures.

CO7: Implement simple decentralized voting or token systems.

Assignment No.	Assignment Details
1	Implement a simple hashing function (SHA-256) in JavaScript to demonstrate data immutability.
2	Create a block structure in JavaScript with index, timestamp, data, previous hash, and current hash.
3	Implement a blockchain ledger in JavaScript that links blocks via hashes.
4	Write a script to validate the integrity of the blockchain (detect tampering).
5	Implement a simple Proof-of-Work (PoW) consensus algorithm (mining) in JavaScript.
6	Set up Remix IDE and deploy a "Hello World" smart contract.
7	Create a Solidity contract to manage user account balances.
8	Implement a state variable and functions to modify state in Solidity.
9	Develop a smart contract that implements mapping for user data storage.
10	Create a contract using modifier for access control (e.g., restricted access).
11	Install and configure MetaMask to connect to a testnet (e.g., Sepolia).
12	Deploy a smart contract on a testnet using Remix and MetaMask.
13	Create a simple HTML/JS frontend to interact with a deployed smart contract.
14	Develop a Decentralized Voting application smart contract.
15	Implement a basic ERC-20 token contract and test transfers.

CO-PO Mapping Table:

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PO13
CO1	2	3	1	3	2	1	2	2	3	1	1	2	1
CO2	3	3	2	3	3	2	2	3	3	1	2	2	2
CO3	3	3	3	3	3	2	3	3	3	2	2	3	2
CO4	2	3	2	2	2	2	2	3	3	1	1	2	1
CO5	3	3	3	3	2	3	2	3	3	2	2	3	3
CO6	3	3	2	3	3	2	3	3	3	1	1	2	1
CO7	2	3	2	3	3	2	2	2	3	1	2	3	3

Weight: 1 - Partially related 2 - Moderately Related 3 - Strongly related

Justification

PO1: Development of blockchain structures, smart contracts, and dApps ensures strong conceptual and applied knowledge of decentralized systems.

PO2: All COs emphasize hands-on implementation (hashing, smart contracts, deployment, dApps), strongly aligning with professional practice.

PO3: Smart contract development and decentralized application/voting systems strongly support innovation and entrepreneurial application.

PO4: Cryptographic hashing, Solidity programming, blockchain analysis, and system implementation build specialized technical competencies.

PO5: Designing consensus mechanisms, analyzing transactions, and implementing decentralized systems require strong analytical and problem-solving ability.

PO6:dApp interaction and decentralized system implementation encourage collaboration, technical discussion, and presentation of solutions.

PO7: Security analysis, smart contract deployment, and transaction evaluation promote investigative and research-oriented learning.

PO8: Continuous exploration of Ethereum tools, Solidity, and decentralized systems supports self-learning and adaptability.

PO9: All outcomes require active use of blockchain platforms, wallets, test networks, and programming tools, strongly supporting digital proficiency.

PO10: Decentralized systems promoting transparency and inclusivity moderately relate to multicultural and ethical collaboration contexts.

PO11: Understanding secure, transparent blockchain applications moderately supports ethical awareness and responsible technology use.

PO12: Independent smart contract deployment and system implementation require ownership, responsibility, and accountability.

PO13: Implementing decentralized voting/token systems strongly connects to community-oriented and service-based blockchain applications.