Nirmala Memorial Foundation College of Commerce and Science

(Autonomous)

Re-accredited by NAAC with B++, ISO 9001-2015 Certified, Recognised under section 2(f) & 12(B) of the UGC Act 1956

Permanently Affiliated to the University of Mumbai,

Syllabus

Bachelor of Science in Computer Science

(B.Sc. Computer Science)

Semester III and IV

(Programme Code: UGCS02)

As Per New Education Policy (NEP) 2020

(To be implemented from the Academic Year 2026-2027)

Approved in the Academic Council Meeting held on ***8th June 2026***

INTRODUCTION OF THE PROGRAMME

1. Introduction

The **Second Year** of the **B.Sc. Computer Science** program serves as a pivotal phase in the academic journey of students, deepening their understanding of both theoretical and applied aspects of computing. This stage is structured to strengthen their analytical thinking and programming abilities while introducing them to essential components of system-level programming, computational theory, and software development life cycles.

Core subjects such as **Data Structures, Operating Systems**, and the **Theory of Computation** form the backbone of this year, equipping students with skills to design efficient algorithms, understand system architecture, and explore the mathematical foundations of computing. Simultaneously, practical programming knowledge is enhanced through courses like **Java Programming**, which emphasizes object-oriented programming and GUI development, and **Computer Networks**, which provides an insight into data transmission, network protocols, and communication models.

An important aspect of this year is the inclusion of **Software Engineering** and **IoT Technologies**, which bridge theory with real-world application by guiding students through the structured development of software and embedded systems. Further, skill-oriented courses like **Mobile Application Development** and **MEAN Stack Development** expose students to industry-relevant tools and frameworks, fostering full-stack development and innovation.

Overall, this phase of the program is designed to shape students into competent problem solvers, capable of building scalable software solutions and understanding the internal workings of computing systems. It also encourages critical thinking, creativity, and a readiness to explore emerging technologies, laying a solid foundation for their final year of specialization and future career paths.

2. Aims and Objectives

Deepen Conceptual Understanding: To develop a strong conceptual understanding of intermediate level topics such as Operating Systems, Theory of Computation, Computer Networks, and Software Engineering.

Hands-on Skill Development: To provide hands-on experience through practical labs in Data Structures, Java Programming, Mobile App Development, and IoT, encouraging students to apply their learning to solve real-world problems.

Design & Analytical Skills: To strengthen algorithmic and analytical skills, enabling students to design efficient data structures, evaluate system performance, and understand formal models of computation.

Software Project Exposure: To introduce students to structured software development processes, including requirement analysis, design, testing, and documentation using Software Engineering principles.

Technology Familiarity: To familiarize students with industry-relevant technologies such as MEAN stack and IoT to encourage innovation and interdisciplinary application.

Teamwork & Communication: To promote collaborative project work and improve communication skills through documentation and presentations, essential for industry-readiness

PROGRAM OUTCOME

At the end of three year Bachelor of Computer Science the students will be able:

- Formulate, model, and design solutions and procedures, utilizing software tools to address real-world problems effectively.
- Design and develop computer programs and computer-based systems in diverse areas such as networking, web design, security, cloud computing, IoT, data science, and other emerging technologies.
- Familiarize themselves with modern-day trends in industry and research-based settings, fostering the ability to innovate novel solutions to existing problems.
- Apply concepts, principles, and theories related to computer science to new and challenging situations.
- Demonstrate proficiency in using current techniques, skills, and tools essential for computing practice.
- Apply standard Software Engineering practices and strategies in real-time software project development.
- Pursue higher studies of specialization and confidently enter technical employment.
- Work independently or collaboratively as effective team members on substantial software projects, showcasing project management and teamwork skills.
- Communicate and present their work effectively and coherently, both in oral and written formats.
- Display ethical conduct in the usage of the Internet and Cyber systems, understanding and adhering to ethical standards in computing practices.
- Engage in independent and life-long learning, adapting to the rapidly changing IT industry and staying abreast of evolving technologies.

SEMESTER III

COURSE STRUCTURE

VERTICAL		COURSE CODE	SUBJECT	CREDIT
Vertical 1	Major	NUCS301	Principles of Operating Systems	**2**
	Major	**NUCS302**	**Software Engineering**	**2**
	Major	NUCS303	Data Structures	**2**
	Major	NUCS304	Computer Science Practical 3 (PR)	**2**
Vertical 2	Minor	NUCS305	DS_Python for Data Science	**2**
	Minor	NUCS306	DS_Scala for Data Science	**2**
Vertical 3	OE1	NUCS307	Financial Literacy	**2**
Vertical 4	VSC	NUCS308	Java Programming	**2**
Vertical 5	AEC	NUHN301	Hindi Bhasha: Vyavaharik Prayog	**2**
Vertical 6	CC	NUCC303	Co-curricular	**2**
	FP	NUFP301	Field Project	**2**
			Total Credits	**22**

SEMESTER IV

COURSE STRUCTURE

VERTICAL		COURSE CODE	SUBJECT	CREDIT
Vertical 1	Major	NUCS401	Computer Networks	**2**
	Major	**NUCS402**	**Agile Software Development and DevOps**	**2**
	Major	NUCS403	IoT Technologies	**2**
	Major	NUCS404	Computer Science Practical 4	**2**
Vertical 2	Minor	NUCS405	DS_Advanaced Python for Data Science	2
	Minor	NUCS406	DS_Data Analysis with SAS/SPAA/R	2
Vertical 3	OE	NUCS407	Digital Payment Systems and Security	2
Vertical 4	SEC	NUCS408	Mobile Application Development	2
Vertical 5	AEC	NUEN402	Hindi Bhasha : Kaushal Ke Aadhar	**2**
Vertical 6	CC	NUCC403	Co-curricular	2
	CEP	NUCP401	Community Engagement Program	**2**
			Total Credits	**22**

Semester – III

Vertical – I

Mandatory Courses

Name of the Course: Principles of Operating Systems – NUCS301

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Principles of Operating Systems course introduces students to the fundamental role an operating system plays in managing hardware and software resources. It covers essential concepts like process management, memory handling, file systems, and CPU scheduling, providing a strong foundation in how computers function at a low level. **Relevance:** This course is highly relevant as it bridges the gap between computer hardware and user applications. Understanding the operating system's role helps students grasp how various programs run efficiently and how resources like memory and CPU are allocated and managed. **Usefulness:** The course equips students with practical knowledge that is directly applicable in configuring, using, and troubleshooting different operating systems such as Linux, Windows, and macOS. It also deepens their understanding of how applications interact with system resources, which is critical for developers, testers, and IT professionals. **Application:** Operating system principles are used extensively in fields such as embedded systems, mobile application development, cloud computing, and cybersecurity. From smartphones to servers and IoT devices, the knowledge of OS design and implementation is central to building and maintaining modern computing environments. **Interest:** This course excites students by uncovering the inner workings of the systems they use every day. The hands- on aspects, like working with commands, simulating scheduling algorithms, and exploring file structures, make it both intellectually engaging and practically rewarding. **Connection with Other Courses:** The concepts learned here support and are reinforced in several other subjects such as Computer Architecture, Data Structures & Algorithms, System Programming, and Cybersecurity. The interdisciplinary nature of the course strengthens students' overall understanding of the computer science domain. **Demand in the Industry:** There is a consistent demand in the IT industry for professionals who understand the workings of operating systems. Tech companies developing system software, embedded platforms, or managing cloud infrastructure seek candidates who have a strong grasp of OS-level functioning.

		Job Prospects: Career opportunities after learning this subject include roles like system administrator, kernel developer, embedded systems engineer, and DevOps engineer. A deep knowledge of operating systems also opens doors to specialized fields such as cybersecurity, performance tuning, and system software development.
2	**Vertical:**	**Major**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. To learn basic concepts and structure of operating systems 2. To understand process communication techniques 3. To study various CPU scheduling algorithms 4. To learn about Memory management 5. To learn about File system management and implementation	
8	**Course Outcomes:** After successful completion of this course, students would be able to - **CO 1.** Work with any type of operating system **CO 2.** Handle threads, processes, process synchronization **CO 3.** Implement CPU scheduling algorithms **CO 4.** Understand the background role of memory management **CO 5.** Design file system	
9	**Modules:-** **Module 1 (15 hours):**	
	Fundamentals of Operating systems – Definition of Operating System, Operating System Operations, Functions of Operating System, Generation of Operating System,Types of Operating System, Fault tolerance, Operating System as a User Interface, System Calls, Types of System Calls, Operating-System Structure, Overview of different operating system **Processes –** Threads - Overview, Multicore Programming, Multithreading Models, Process Concept, Process Scheduling, Operations on Processes, Inter-process Communication **Process Synchronization –** General structure of a typical process, race condition, The Critical-Section Problem, Mutex Locks, Semaphores, Classic Problems of Synchronization **CPU Scheduling –** Basic Concepts, Scheduling Criteria, Scheduling Algorithms (FCFS, SJF, SRTF, Priority, RR, Multilevel Queue Scheduling), Thread Scheduling	
	Module 2 (15 hours):	
	Deadlocks – System Model, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock **Memory Management –** Main memory background, Logical address space, Physical address space, MMU, Swapping, Contiguous Memory Allocation, Segmentation, Paging	

	Virtual Memory – Background, Demand Paging, Copy-on-Write, Page Replacement, Allocation of Frames,Mass-Storage Structure: Overview, Disk Structure, Disk Scheduling, Disk Management **File System Interface and Implementation –** File Concept, Access Methods, Directory and Disk Structure, File-System Mounting, File Sharing File-System Structure, File-System Implementation
10	**Text Books** 1. Abraham Silberschatz, Peter Galvin, Greg Gagne, Operating System Concepts, Wiley, 9th Edition
11	**Reference Books** 1. Achyut S. Godbole, Atul Kahate, Operating Systems, Tata McGraw Hill 2. Naresh Chauhan, Principles of Operating Systems, Oxford Press 3. Andrew S Tanenbaum, Herbert Bos, Modern Operating Systems, 4e Fourth Edition, Pearson Education, 2016
12	**Internal Continuous Assessment: 40%** / **Semester End Examination: 60%**
13	**Continuous Evaluation through:** Class test of 15 marks Quizzes/ Presentations/ Assignments: 5 marks Total: 20 marks **Format of Question Paper: External Examination (30 Marks)– 1 hr duration**
14	**Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)**

Questions	Based On	Options	Marks
Q.1	Module 1	Any 2 out of 4	10
Q.2	Module 2	Any 2 out of 4	10
Q.3	Module 1 & 2	Any 2 out of 4	10

Name of the Course: Software Engineering - NUCS402

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** Software Engineering is the foundation for systematic software development. It introduces learners to the lifecycle of software— from planning and modeling to development and testing. The course provides a disciplined approach to creating software that meets user needs and performs reliably under real-world conditions. **Relevance:** In today's digital age, software is integral to almost every sector. This subject is crucial as it prepares students to handle increasing software complexity and ensures they understand the importance of process driven development and quality standards. **Usefulness:** The course equips students with practical tools and methodologies for planning, estimating, designing, and testing software. It fosters analytical thinking and technical communication, which are vital for handling real-life software projects effectively and efficiently. **Application:** Software Engineering principles are applied in various domains— banking, healthcare, education, mobile app development, and more. The skills learned here help in managing full-scale projects, ensuring user satisfaction and product scalability. **Interest:** The course offers engaging content like UML modeling, agile methods, risk management, and testing strategies. Students enjoy applying these concepts through diagrams, real-world case studies, and collaborative project planning. **Connection with Other Courses:** This subject ties well with courses in Object-Oriented Programming, Database Management, and Final Year Projects. It bridges theoretical knowledge with project execution skills, enhancing overall technical competence. **Demand in the Industry:** There is a high demand for software professionals who understand both development and project management. Agile development, DevOps, and software quality assurance are key skills sought by employers globally. **Job Prospects:** Completing this course opens doors to roles such as software developer, quality analyst, system designer, project assistant, and scrum team member. With experience, one can grow into roles like project manager, solution architect, or process consultant.

2	**Vertical:**	**Major**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Provide foundational knowledge of software engineering processes, development life cycle models, and methodologies. 2. Introduce software requirements analysis and system modeling using tools like UML. 3. Explain key software design principles, project estimation, scheduling, and risk management techniques. 4. Familiarize students with software quality assurance, configuration management, and software testing techniques.	
8	**Course Outcomes :** After successful completion of this course, students will be able to: **CO 1.** Explain and apply appropriate software process models to various project scenarios. **CO 2.** Analyze requirements and develop system models using UML. **CO 3.** Apply software design principles, estimation, and scheduling techniques. **CO 4.** Demonstrate knowledge of quality assurance and implement effective software testing strategies.	
9	**Modules:-** **Module 1 (15 hours):**	
	Introduction to Software Engineering: Nature of Software, Software Engineering: Principles and Practice, Software Process Framework, Layered Technology, Process Framework, Process Patterns, Capability Maturity Model (CMM), Process Assessment **Software Development Models:** Prescriptive Models: Waterfall, Incremental, Rapid Application Development (RAD), Evolutionary Models: Prototyping, Spiral Model, Specialized Models: Component-Based Development, Aspect-Oriented Development, Agile Development: Agile Manifesto, Extreme Programming (XP), Scrum Overview **Requirements Engineering and Analysis:** Requirements Engineering Process, Elicitation Techniques (Interviews, Workshops, Use Cases), Components & Characteristics of a Software Requirements Specification (SRS), Validation of Requirements **System and Object-Oriented Modeling (using UML):** Use Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram, State Chart Diagram, Component & Deployment Diagram	
	Module 2 (15 hours):	
	Software Design and Architecture: Design Principles: Coupling and Cohesion, Functional-Oriented vs. Object-Oriented Design, System Architecture Design, Design Verification and Validation, Monitoring and Control in Design **Software Metrics and Estimation:** Software Measurement: LOC, Function Point, and Use Case-Based Estimations, Object-Oriented Metrics, Empirical Estimation Models,	

	Introduction to COCOMO II, Estimation in Agile Development, Make/Buy Decision **Software Project Management:** Project Planning: Scope, Feasibility, Resource Estimation, Project Scheduling: Effort Estimation, Time-Line Charts, Gantt Charts, Risk Management: Identification, Projection, RMMM Plan **Software Quality Assurance & Testing:** SQA Activities, Software Reviews, Formal Technical Reviews (FTR), Software Reliability and SQA Plan, Verification & Validation, Testing Principles and Objectives, Test Oracles, Levels of Testing, White box and Black-box Testing, Test Plan and Test Case Design
10	**Text Books** 1. Software Engineering, A Practitioner's Approach, Roger S, Pressman, 2019 2. Software Engineering: principles and Practices, Deepak Jain, OXFORD University Press, 2008
11	**Reference Books** 1. Software Engineering, Ian Sommerville, Pearson Education, 2017 2. Fundamentals of Software Engineering, Fourth Edition, Rajib Mall, PHI, 2018 3. Software Engineering: Principles and Practices, Hans Van Vliet, John Wiley & Sons, 2010 4. A Concise Introduction to Software Engineering, Pankaj Jalote, Springer
12	**Internal Continuous Assessment: 40%** / **Semester End Examination: 60%**
13	**Continuous Evaluation through:** Class test of 15 marks Quizzes/ Presentations/ Assignments: 5 marks Total: 20 marks **Format of Question Paper: External Examination (30 Marks)– 1 hr duration**
14	**Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)**

Questions	Based On	Options	Marks
Q.1	Module 1	Any 2 out of 4	10
Q.2	Module 2	Any 2 out of 4	10
Q.3	Module 1 & 2	Any 2 out of 4	10

Name of the Course: Data Structures – NUCS303

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Data Structures course introduces students to the foundational building blocks of programming and software development. It covers the systematic organization, management, and storage of data for Efficient access and modification using various data structures. **Relevance:** Data structures are essential for solving computational problems efficiently, making this subject central to the study of computer science. Understanding how data is stored, manipulated, and accessed directly impacts the performance of applications. **Usefulness:** Knowledge of data structures allows students to choose the most appropriate structure for any problem, enabling optimal algorithm design and effective memory utilization. This leads to better program performance and scalability. **Application:** Data structures find wide applications in database indexing, compiler design, network routing, artificial intelligence, and graphics. Real-life scenarios such as job scheduling, expression evaluation, searching, and pathfinding extensively use stacks, queues, trees, and graphs. **Interest:** Students often find this subject intellectually stimulating as it challenges them to think logically and solve problems efficiently. Implementing structures like AVL Trees or graph traversals can be deeply engaging and rewarding. **Connection with Other Courses:** This course is closely linked with algorithms, operating systems, databases, and programming languages. Many advanced subjects assume a working knowledge of data structures, making it a prerequisite for deeper computer science learning. **Demand in the Industry:** Almost every tech role—from software development to system design—requires a strong understanding of data structures. Employers frequently test these concepts during technical interviews and coding assessments. **Job Prospects:** Proficiency in data structures opens doors to careers in software engineering, data science, system architecture, web development, and cybersecurity. It provides a solid base for roles that involve designing efficient and scalable solutions.
2	**Vertical:**	**Major**
3	**Type:**	**Theory**

4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Introduce the concept of Abstract Data Types (ADTs) and fundamental data structures for efficient data representation and manipulation. 2. Develop an understanding of linear data structures such as linked lists, stacks, and queues, along with their applications. 3. Explore non-linear data structures like trees and graphs and their use in solving real-world problems. 4. Understand advanced structures such as priority queues, heaps, and hashing for efficient data access and organization.	
8	**Course Outcomes:** After successful completion of this course, students will be able to: **CO 1.** Define and implement linear and non-linear data structures using Abstract Data Types (ADTs) for efficient data handling. **CO 2.** Perform operations on linked lists, stacks, and queues, and apply them to real world problems like expression evaluation and scheduling. **CO 3.** Design and traverse tree structures such as binary trees and AVL trees, and apply them in encoding, searching, and organizing data. **CO 4.** Implement graph algorithms and advanced structures like heaps and hashing for solving connectivity, priority, and lookup problems.	
9	**Modules:-** **Module 1 (15 hours):**	
	Abstract Data Type: Different Data Types, different types of data structures & their classifications, Introduction to ADT, Creating user-specific ADT **Linked Structures**: ADT for linked list, Advantages & Disadvantages, Singly Linked List-Traversing, Searching, Prepending and Removing Nodes, applications of linked list like polynomial equation, ADT of doubly linked list, Advantages & Disadvantages, Insertion and deletion of nodes at various positions **Stacks**: Stack ADT for Stack, Advantages & Disadvantages, Applications of stack like balanced delimiter, prefix to postfix notation **Queues**: Queue ADT, Advantages & Disadvantages, linked representations. Circular Queue operations, Dequeues, applications of queue like job scheduling queues	
	Module 2 (15 hours):	
	Trees: ADT for Tree Structure. Advantages & disadvantages, Binary Tree-Properties, Implementation and Traversals, Binary Search Tree, Balanced BST, Threaded Binary Trees, AVL Trees, Applications of Tree like Huffman Coding, **Priority Queues & Heaps:** Priority Queue, Priority Queue ADT, Advantages and Disadvantages, Applications, Heaps, types of heaps, Heapifying the element, **Graph:** Introduction, Graph ADT, Advantages and Disadvantages, Graph Representation using adjacency matrix and adjacency list, Graph operations like insertion and deletion of nodes, Graph Traversals using BFS & DFS, Applications of Graphs like shortest path algorithms, **Hashing:** Hash Table ADT, Advantages & Disadvantages, Concept of hashing, hash	

	table, hash functions, collision, collision avoidance techniques, Applications of hashing
10	**Text Books** 1. Introduction to Algorithm, Thomas H Cormen, PHI 2. Data Structures And Algorithms Made Easy, Narasimha Karumanchi, 2021
11	**Reference Books** 1. Fundamentals of Computer Algorithms, Sartaj Sahni and Sanguthevar Rajasekaran Ellis Horowitz, Universities Press, 2018 2. Data Structures and Algorithms in Python, Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, Wiley, 2016

12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Class test of 15 marks Quizzes/ Presentations/ Assignments: 5 marks Total: 20 marks	**Format of Question Paper: External Examination (30 Marks)– 1 hr duration**

14 **Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)**

Questions	Based On	Options	Marks
Q.1	Module 1	Any 2 out of 4	10
Q.2	Module 2	Any 2 out of 4	10
Q.3	Module 1 & 2	Any 2 out of 4	10

Name of the Course: Computer Science Practical 3 – NUCS304

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** This course is a practical extension of theoretical concepts covered in Operating Systems and Data Structures. It introduces students to essential programming constructs required for system-level programming and structured data organization. Through carefully curated lab exercises, students gain insights into concurrency, resource management, scheduling, and structured data processing. **Relevance:** Operating Systems and Data Structures form the foundation of all computing systems and applications. Understanding how data is stored, accessed, manipulated, and how system resources are scheduled and synchronized is vital for every computer science graduate. This practical course equips students with essential skills that are applicable across all domains of software development and system programming. **Usefulness:** This course builds a strong foundation in system-level programming and abstract data handling, essential for any computer science graduate. It enhances the ability to understand and implement core concepts like scheduling, synchronization, and data abstraction through real-time coding tasks. Students gain valuable skills for debugging, memory management, and efficient algorithm design. These skills are crucial for both academic excellence and industry readiness. **Application:** The practical skills developed in this course are directly applicable in the design of operating system modules, file systems, and memory management tools. Data structures like trees, heaps, and graphs are widely used in building compilers, databases, and networking software. The understanding of multi-threading and process synchronization enables students to contribute to applications requiring concurrency, such as games, real- time simulations, and cloud systems. These applications form the backbone of modern software development. **Interest:** Students often find this course engaging due to its interactive and logic-based approach to solving real-world problems. Concepts like circular queues for task scheduling or graphs for social network analysis spark curiosity and hands-on involvement. The implementation of real-time synchronization and visual data traversals makes the learning process both stimulating and rewarding. This course encourages creative problem-solving through code. **Connection with Other Courses:** This practical course closely aligns with theoretical subjects like Operating Systems, Data Structures and Algorithms, and Object-Oriented Programming. It also forms the basis for advanced topics such as Distributed Systems, Artificial Intelligence, and Systems Programming. Concepts learned here are frequently reused and expanded upon in courses like Database Systems, Compiler Design, and Software Engineering. Hence, it serves as a vital link across the curriculum.

		Demand in the Industry: Industry consistently seeks professionals proficient in data structures and operating system fundamentals. Skills like process synchronization, memory optimization, and graph traversal are essential for roles in software development, cloud computing, and backend engineering. These competencies are tested during technical interviews at leading firms, including product-based and service based companies. The ability to translate theoretical knowledge into efficient code is highly valued across tech domains. **Job Prospects:** Completing this course prepares students for roles such as Software Developer, System Programmer, Backend Engineer, or Technical Consultant. It opens opportunities in IT services, product development, and tech startups. With further specialization, students can also pursue careers in AI/ML, cybersecurity, and data engineering. The practical exposure provided here serves as a strong launch pad for core technical positions in the industry.
2	**Vertical:**	**Major**
3	**Type:**	**Practical**
4	**Credits:**	**2 credits(1 credit = 30 Hours of Practical work in a semester)**
5	**Hours Allotted:**	**60 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Develop practical skills in implementing core concepts of operating systems & data structures. 2. Simulate real-world scenarios involving process management, synchronization, and memory management. 3. Strengthen understanding of data abstraction using linked structures, trees, graphs, and hashing techniques. 4. Enhance problem-solving through analysis, implementation, and evaluation of algorithmic strategies in system-level and data-driven programs.	
8	**Course Outcomes:** After successful completion of this course, students will be able to: **CO 1.** Design and implement solutions using inter-process communication, synchronization, and scheduling techniques. **CO 2.** Construct and manipulate linear and non-linear data structures for solving computational problems. **CO 3.** Apply stacks, queues, trees, graphs, and hash tables effectively in algorithm and application development. **CO 4.** Analyze and evaluate memory, disk management, and data operations to build efficient system-level applications.	
9	**Modules:-** **Module 1 : Practical based on Principles of Operating Systems (30 hours):**	
	Process Communication using Shared Memory • Implement producer-consumer synchronization using shared memory and semaphores. **Process Communication using Message Passing**	

- Use message queues/pipes to solve the producer-consumer problem.

Threading and Single Thread Control Flow

- Practice thread creation and basic thread lifecycle using standard libraries (e.g., pthreads or Java threads).

Multi-threading and Fibonacci Generation

- Implement multi-threading to generate and print Fibonacci sequences.

Process Synchronization and Bounded Buffer Problem

- Implement buffer control with synchronized access.

Readers-Writers Problem – Synchronization in Shared Access

- Implement a solution using semaphores that prioritizes readers or writers, allows multiple readers or a single writer at a time, ensures fair access, and prevents deadlock.

CPU Scheduling Algorithms (Part 1) – FCFS and Non-preemptive Scheduling

- Implement First-Come, First-Serve (FCFS) scheduling, extend it to general non preemptive scheduling, and analyze waiting time, turnaround

CPU Scheduling Algorithms (Part 2) – Round Robin

- Implement Round Robin scheduling with configurable time quantum.

Memory Management Techniques

- Simulate FIFO and LRU page replacement using page reference strings.

Disk Scheduling and Simple File System Design

- Simulate FCFS, SSTF, C-SCAN, C-LOOK, RSS for disk head movement · Design a basic file system structure with block allocation, directory management, and file operations (create, read, delete).

Module 2: Practical based on Data Structures (30 hours)

Exploring Abstract Data Types (ADT) & Custom Structures

- Create and manipulate structures to model ADTs like Student, Book, or Employee.
- Implement basic operations (create, update, delete) using structures.
- Reflect on differences between primitive and abstract data types.

Building and Using Singly Linked Lists

- Construct a dynamic singly linked list with basic operations.
- Apply linked lists to simulate scenarios such as managing a playlist or to-do list.
- Compare static (array) vs dynamic (linked) approaches.

Polynomial Operations Using Linked Lists

- Represent polynomials using linked lists.
- Perform polynomial addition and subtraction by merging lists.
- Use structured representation to reinforce node manipulation.

Working with Doubly Linked Lists

- Create a doubly linked list with forward and backward traversal.
- Implement insertion/deletion at head, tail, and specific positions.
- Use in scenarios like browser history or undo-redo features.

Implementing and Using Stack ADT

- Implement push, pop, peek using arrays or linked lists.
- Solve problems like delimiter matching or undo mechanism.
- Convert expressions from prefix to postfix and evaluate them.

Understanding Queues and Circular Queues

- Develop linear and circular queues to simulate task scheduling.

	• Perform enqueue and dequeue with wrap-around logic. • Discuss memory utilization in linear vs circular queues. **Tree Traversals and Binary Search Trees** • Create a binary search tree (BST) from a dataset. • Perform and visualize in-order, pre-order, and post-order traversals. • Use traversal results to derive sorted sequences. **Balanced Trees & Priority Queues** • Insert values and observe AVL tree rebalancing. • Construct min-heaps or max-heaps and simulate priority queues. • Use priority queues to manage task priorities (e.g., patient triage, job scheduling). **Graph Representations and Traversals** • Represent graphs using adjacency matrices and lists. • Implement BFS and DFS to explore graph components. • Use graphs for mapping routes or exploring social networks. **Hashing Concepts and Collision Handling** • Implement a hash table with chaining or linear probing. • Simulate insertion, deletion, and search with collisions. • Discuss practical hashing applications (e.g., dictionary lookup, indexing).
10	**Text Books** 1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2022). Operating system concepts (10th ed.). Wiley. 2. Aho, A. V., Ullman, J. D., & Lam, M. S. (2021). Data Structures and Algorithms in Python (Adapted by R. Rao). Pearson India.
11	**Reference Books** 1. Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2014). Data Structures and Algorithms in Java (6th ed.). Wiley India. 2. Kanetkar, Y. (2020). Data Structures Through Python (1st ed.). BPB Publications.
12	**Internal Continuous Assessment: 40%** / **Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments • Journal Submission : 10 Marks • Assignments: 10 Marks Total: 20 marks **A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination : 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)

Vertical – II

Minor Courses

Name of the Course: Python for Data Science – NUCS305

Sr. No.	Heading	Particulars
1	**Description the course:**	Advanced python programming practical modules make able to acquire knowledge for implementing python code for various applications such as handling data, analysing and visualizing data. Database Management System's practical approach is useful to gain the knowledge for software backend development. It benefits to user by providing data definition, data access, reduced data redundancy, data integrity, data sharing, data organizing, data consistency, data accuracy, and security.
2	**Vertical:**	**Minor**
3	**Type:**	**Practical**
4	**Credits:**	**2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Implement Python for Data Processing – Utilize tuples, regular expressions, date-time functions, and libraries like NumPy and Pandas for data manipulation. 2. Understand Relational Databases & SQL – Identify entities, relationships, and relational structures while implementing constraints using SQL. 3. Perform Data Retrieval & Manipulation in SQL – Execute DML operations, apply built-in functions, retrieve and aggregate data, and work with joins and nested queries. 4. Manage Database Security & Access Control – Implement user access controls, security measures, and database backup strategies.	
8	**Course Outcomes:** After successful completion of this course, students would be able to - CO1. Apply Python for Data Handling – Utilize lists, tuples, regular expressions, date-time functions, and libraries like NumPy and Pandas for data processing. CO2. Execute SQL Queries for Data Operations – Perform CRUD (Create, Read, Update, delete) operations, table modifications, and database backup/restoration using SQL. CO3. Retrieve & Analyze Data Using SQL – Use aggregate functions, joins, and nested queries to extract meaningful insights from relational databases. CO4. Manage Database Security & Optimization – Implement access control, create virtual tables, and optimize database structures for secure and efficient data management.	
9	**Modules:-** **Module 1 (15 hours):**	
	Practical 1: Concept: Tuples and Date-Time Functions • Write a Python program to create nested tuples, access elements, and unpack values.	

	• Use datetime to calculate the difference between two dates (e.g., age calculation). **Practical 2: Concept: Regular Expressions for Pattern Matching** • Develop a Python program that extracts valid email IDs and phone numbers from a given text using the re module. **Practical 3: Concept: Data Analysis using NumPy** • Create a NumPy array from a list. Perform basic operations like sum, mean, median, min, max, and reshaping. **Practical 4: Concept: Data Manipulation using Pandas** • Load data from a CSV file into a DataFrame. • Perform operations like filtering rows, adding new columns, and grouping by a categorical variable. **Practical 5: Concept: Data Analysis using Pandas** • Generate summary statistics (mean, median, standard deviation) for numerical columns. • Export the modified DataFrame to a new CSV file. ***Note : Create a mini assignment for students that covers the above concepts in a single question***
	Module 2 (15 hours):
	Practical 6: Concept: Creating Database and Tables with Constraints • Create a database. • Define tables with appropriate primary keys, foreign keys, unique, and check constraints (e.g., Student and Department tables). **Practical 7: Concept: Inserting, Updating, and Deleting Data** • Insert sample data into tables using INSERT. • Update certain records (e.g., change student address). • Delete a specific record based on a condition. **Practical 8: Concept: Data Retrieval and Aggregate Functions** • Write SQL queries using SELECT, WHERE, ORDER BY. • Use aggregate functions (SUM, AVG, COUNT, MAX, MIN) and GROUP BY. **Practical 9 : Concept: Joins and Nested Queries** • Write queries using INNER JOIN, LEFT JOIN, and RIGHT JOIN. • Write a nested query (e.g., find students who scored above the class average). **Practical 10:Concept: User Access Control and Privileges, Database Backup and Restore** • Create new database users. • Grant and revoke privileges (e.g., SELECT, INSERT, UPDATE). • Demonstrate access with limited rights. • Write SQL commands (or use DBMS utilities) to back up a database. ***Note : Create a mini assignment for students that covers the above concepts in a single question***
10	**Text Books** 2. Programming through Python M. T. Savaliya, R.K Maurya, G.M Magar, Staredu Solutions, 1st edition (2018)

	3. Python DataScience Handbook, Jake VanderPlas, O'Reilly Media, 1st edition (2016)	
11	**Reference Books** 4. Let Us Python, Yashwant Kanetkar, BPB publication , 1st edition (2019) 5. Programming in Python3, Mark Summerfield, Pearson Education, 2nd edition (2018) 6. Learning Python, LutzM, O'Reilly- Shroff, 5th edition, 2013. 7. Beginning Python, Magnus LieHetland, Apress, 2nd edition, 2009. 8. Star Python, Star Certification, Star Certification,1st , 2018.	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments • Journal Submission : 10 Marks • Assignments: 10 Marks Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination : 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)	

Name of the Course: Scala for Data Science – NUCS306

Sr. No.	Heading	Particulars
1	**Description the course:**	This course provides hands-on experience with Scala and its ecosystem for data analysis and machine learning. Students will learn statistical methods, machine learning algorithms, and data processing techniques using Breeze and Apache Spark. The course also covers time-series analysis, feature engineering, and building scalable data pipelines. Through practical exercises, students will gain proficiency in implementing regression models and clustering while handling real-world datasets effectively.
2	**Vertical:**	**Minor**
3	**Type:**	**Practical**
4	**Credits:**	**2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. To set up and configure Scala, SBT, and Apache Spark for programming, data analysis, and large-scale data processing. 2. To perform statistical calculations, including correlation, frequency distribution, and moving averages using Scala and Breeze, and visualize data insights with Breeze-viz. 3. To implement machine learning models such as linear regression, logistic regression, and k-means clustering, along with feature engineering for predictive modeling. 4. To utilize the Breeze library for numerical computations, matrix operations, and time-series data analysis to extract meaningful insights.	
8	**Course Outcomes:** After successful completion of this course, students would be able to - CO1. Students will set up a functional Scala development environment with SBT and execute basic programs for data analysis. CO2. Students will utilize Breeze for numerical operations, matrix manipulations, and statistical computations such as correlation and frequency distribution. CO3. Students will create data visualizations using Breeze-viz and implement machine learning models, including regression and clustering, using Breeze. CO4. Students will work with Apache Spark for large-scale data processing, machine learning pipelines, and time-series analysis to extract meaningful insights.	
9	**Modules:-** **Module 1 (15 hours):**	
	Set up Scala and SBT on your system. 1. Write a simple Scala program that prints a welcome message for data scientists. 2. Calculate mean, median, and mode of a list of numbers. Implement basic statistical	

	calculations using Scala collections. 3. Generate a random dataset of 10 numbers and calculate its variance and standard deviation. 4. Create a dense vector using Breeze and calculate its sum, mean, and dot product with another vector. 5. Generate a random matrix using Breeze and compute its transpose and determinant. 6. Slice a Breeze matrix to extract a sub-matrix and calculate its row and column sums. 7. Write a program to perform element-wise addition, subtraction, multiplication, and division of two Breeze matrices. 8. Read a CSV file and calculate basic statistics for each numeric column. Use the scala-csv library or similar tools. 9. Handle missing values in a dataset. Replace missing values with the column mean. 10. Filter rows in a dataset where a specific column value exceeds a threshold. 11. Write a program to tokenize and count the frequency of words in a text file. 12. Implement one-hot encoding for a categorical column in a dataset. 13. Create a scatter plot of random data using Breeze-viz. Label the axes and customize the color of points. 14. Generate a histogram of a dataset using Breeze-viz. Experiment with different bin sizes. 15. Plot a line graph for a dataset showing a trend over time. 16. Combine two plots (e.g., scatter and line plot) in a single visualization using Breeze-viz.
	Module 2 (15 hours):
	1. Find the correlation between two lists of numbers. Implement the formula for Pearson correlation coefficient. 2. Calculate the moving average of a time series data using Scala collections. 3. Write a program to compute frequency distribution and cumulative frequency of a dataset. 4. Sort a dataset by a specific column and extract the top 5 rows. 5. Implement linear regression using Breeze. Fit a model to a small dataset and predict a value. 6. Perform logistic regression using Breeze. Classify a dataset with binary labels. 7. Compute the Euclidean distance between two Breeze vectors. Use it for nearest neighbor classification. 8. Cluster a dataset into two groups using k-means clustering in Breeze. 9. Set up Apache Spark locally and count the frequency of words in a text file. 10. Filter rows in a CSV file using Spark DataFrames where a numeric column exceeds a threshold. 11. Perform a group-by operation in Spark DataFrames to compute the average of each group. 12. Join two CSV files in Spark DataFrames based on a common column and write the output to a file.

	13. Create a simple Spark MLlib pipeline to classify data. Use logistic regression or decision trees. 14. Perform basic time series analysis in Scala. Generate synthetic time series data (e.g., daily sales over a month). 15. Create polynomial features from a dataset. Given a list of numbers (e.g., [1, 2, 3]), generate polynomial features up to degree 3 (e.g., [1, 1^2, 1^3, 2, 2^2, 2^3, 3, 3^2, 3^3]).	
10	**Text Books** 1. Scala for Data Science, by Pascal Bugnion, Packt Publishing, 1st edition (28 January 2016) 2. Mastering Scala by Dennis Alexander, Packt Publishing, 1st edition (2023) 3. Scala 3 Mastery by John Hunt, Apress, 1st edition (2023) 4. Mastering Scala 3 by John Hunt, Apress, 1st edition (2023)	
11	**Reference Books** 1. Programming Scala by Dean Wampler and Alex Payne, O'Reilly Media, 3rd edition (2021) 2. Scala Cookbook by Alvin Alexander, O'Reilly Media, 2nd edition (2021) 3. Functional Programming in Scala by Paul Chiusano and Rúnar Bjarnason, Manning Publications, 2nd edition (2023)	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments • Journal Submission : 10 Marks • Assignments: 10 Marks Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination: 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)	

Vertical – III

Open Elective

Name of the Course: Financial Literacy – NUCS307

Sr. No.	Heading	Particulars
1	**Description the course:**	This course provides a comprehensive introduction to the Indian financial system, offering foundational knowledge of its structure, components, and functioning. It delves into the role of banks and other financial institutions in economic development and explores various banking services available to individuals and businesses. The course also introduces students to the basics of insurance and personal financial planning, enabling them to make informed financial decisions and develop sound saving and investment strategies. Through real-world examples, case studies, and practical insights, students will gain a deeper understanding of how financial instruments, services, and institutions interact to support the financial stability of individuals and the economy.
2	**Vertical:**	**Open Electives**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Familiarity with different aspects of financial literacy such as savings, investment, taxation, and insurance. 2. Understand the relevance and process of financial planning. **3.** To promote financial well-being.	
8	**Course Outcomes (CO):** **After completing this course, students will have the knowledge and skills to** • **CO1:** Learners will develop proficiency for personal and family financial planning. • **CO2:** Learners will develop the ability to analyse banking and insurance products. • **CO3:** Learners will Apply the concept of investment planning.	
•	**Modules:-** **Module 1 (10 hours): Financial Planning and Financial Products Introduction** A) Introduction to Saving, Management of spending and financial discipline: Meaning and importance of saving, theoretical principles of financial discipline, Impact of inflation and consumption patterns on personal finance. **B)** Banking products and services, Digitization of financial transactions: Types of bank accounts (Savings, Current, and Fixed Deposits), Services offered by banks: loans,	

overdrafts, etc. Concept and working of debit cards, credit cards, Introduction to digital banking: Features of Net Banking and UPI and theoretical benefits and limitations of digitization in finance.

Module 2 (10 hours): Investment Planning and Management

A) Investment opportunity and financial products: Overview of asset classes: equity, debt, mutual funds, real estate, and gold. Risk and return theory. Time value of money (conceptual level). Role of SEBI and regulatory environment.

B) Insurance Planning: Basic principles of insurance: risk pooling, premium, and insurable interest. Types of insurance: life vs non-life (health, motor, property).

C) Introduction to basic Tax Structure in India for personal taxation, Aspects of Personal tax planning: Importance of tax planning in personal finance.

10 Text Books

- ✓ Halan, Monika. *Let's Talk Money*, Harper Business
- ✓ Pathak, Bharati V. *The Indian Financial System* (selected basic chapters)
- ✓ RBI's "Financial Literacy" booklets and SEBI's student guides

11 Reference Books

1. Introduction to Financial Planning (4th Edition 2017) - Indian Institute of Banking & Finance.
2. Bhasham, A.L, Wonder that was India: A Survey of the Culture of the Indian Subcontinent Before the Coming of the Muslims. London, Sidgwick and Jackson, 1954'
3. Sinha, Madhu. Financial Planning: A Ready Reckoner July 2017, McGraw Hill.
4. Halan, Monika, Let's Talk Money: You've Worked Hard for It, Now Make It Work for You, July 2018 Harper Business.
5. Pandit, Amar The Only Financial Planning Book that You Will Ever Need, Network 18 Publications Ltd.

12 Internal Continuous Assessment: 40% | **External Assessment : 60%**

13 Continuous Evaluation through:

Sr. No	Particular	Marks
1.	Class Test	15
2.	Class participation	05
	Total	**20**

Semester End Examination (30 Marks)

14	**Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)**

Questions	Type & Module	Marks
Q1.	Objective Question A) Multiple choice questions B) True or False	 08 Marks 07 Marks
Q2.	Theory Question	15 Marks
	OR	
Q2.	Theory Question	15 Marks
	Total	**30**

Vertical – IV

Vocational & Skill Enhancement Courses (VSEC)

Name of the Course: Java Programing – NUCS308

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** Java is one of the most widely-used, robust, and versatile programming languages in the software industry. It follows the object-oriented programming paradigm and is known for its platform independence, thanks to the Java Virtual Machine (JVM). This course introduces students to the foundational and advanced features of Java, covering core concepts, GUI development, multithreading, exception handling, and web application development using Servlets and JSP, and database connectivity through JDBC. **Relevance:** In today's software-driven world, Java is a fundamental language that underpins many enterprise-level, web- based, mobile, and desktop applications. Its syntax and concepts are not only essential for programming in Java but also form the basis for understanding other Languages such as C#, Kotlin, and Scala. Java's relevance is further reinforced by its use in Android development, big data (via tools like Hadoop), and large-scale enterprise applications. **Usefulness:** This course builds strong foundational skills in programming and software design. It enables students to construct modular programs using classes, interfaces, and packages. They also learn to handle exceptions, manage concurrent threads, design user-friendly graphical interfaces, and create dynamic, database-connected web applications using technologies like Servlets and JSP. **Application:** The practical aspect of the course allows students to build desktop tools, interactive GUI applications, and basic web portals. By working with JDBC, Swing, and multithreading, students gain hands-on experience in areas that mirror real-world software development scenarios. **Interest:** Java's simplicity, combined with its rich libraries and real-time problem-solving approach, makes learning engaging and rewarding. Students enjoy working on mini- projects, interactive applications, and seeing the immediate impact of their code, which deepens their interest and confidence in programming. **Connection with Other Courses:** The skills acquired in this course directly support and enhance learning in related subjects like Data Structures, Database Management Systems, Operating Systems, Web Development, and Software Engineering. It also forms a foundation for advanced

		electives in Mobile App Development and Enterprise Computing. **Demand in the Industry:** Java developers are in steady demand across industries such as finance, education, e-commerce, and healthcare. Its robustness, scalability, and extensive community support make it a preferred language for backend systems and enterprise-level applications. **Job Prospects:** After completing this course, students are well-prepared for internships and entry-level roles like Java Developer, Backend Developer, Software Engineer, Web Application Developer, and Android App Developer. Mastery of Java also positions students strongly for advanced certifications and competitive programming opportunities.
2	**Vertical:**	VSC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours of Practical work in a semester)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives:** **1.** Build a solid foundation in Java programming with focus on object-oriented principles and core language features. **2.** Enable modular program development using classes, interfaces, packages, and robust exception handling. **3.** Introduce multithreading and collections for managing concurrency and structured data efficiently. **4.** Develop GUI and web-based applications using Swing, event handling, Servlets, JSP, and JDBC.	
8	**Course Outcomes:** **CO 1.** Apply object-oriented programming principles to develop modular and maintainable Java applications. **CO 2.** Implement exception handling, multithreading, and collections to build robust and efficient programs. **CO 3.** Design interactive desktop applications using Swing and event-driven programming. **CO 4.** Develop dynamic web applications using Servlets, JSP, and JDBC for database connectivity.	

9	**Modules:-** **Module: (15 hours)**
	Java Basics and OOP: History and Features of Java, JVM, JDK, JRE, Java Program Structure, Tokens, Data Types, Operators, OOP Principles: Class, Object, Constructor, this, static, Inheritance, Polymorphism (Overloading/Overriding), Abstraction, Encapsulation, Abstract Classes, Interfaces, super. **Packages & Access Specifiers:** Predefined & user-defined packages, access specifiers **Exception Handling:** Pre-Defined Exceptions, try-catch-finally, throw, throws, custom (user defined) exceptions **Practicals to be covered based on above syllabus::** 1. **Java Basics and Program Structure** **a)** Write a Java program to demonstrate the use of different data types and operators. **b)** Explain the role of JVM, JDK, and JRE with the help of a Java program execution process. 2. **Class, Object, Constructor, this and static** **a)** Write a Java program to create a class `Student` using constructors and the `this` keyword to initialize object data. **b)** Write a Java program to demonstrate the use of `static` variables and methods. 3. **Inheritance, Polymorphism and super** **a)** Write a Java program to demonstrate single inheritance using the `super` keyword. **b)** Write a Java program to demonstrate method overloading and method overriding. 4. **Abstraction, Interfaces and Packages** **a)** Write a Java program using an abstract class to calculate the area of different shapes. **b)** Create a user-defined package and demonstrate the use of access specifiers (`public`, `private`, `protected`, and default). 5. **Exception Handling** **a)** Write a Java program to demonstrate the use of `try-catch-finally` with predefined exceptions. **b)** Write a Java program to create and use a custom (user-defined) exception using `throw` and `throws`.
	Module 2: (15 hours)
	Java Foundation Classes & Swing: JFC Overview, Common Swing Components: JFrame, JPanel, JButton, JTextField, JLabel, Layouts, Event Handling using Delegation Event Model, Adapter classes, ActionListener **JDBC:**

	JDBC Architecture & Drivers, Connecting to DB, Statement, PreparedStatement, ResultSet, Navigating data, ResultSetMetaData, Transactions, Exception handling **Practicals to be covered based on above syllabus::** 1. **Java Foundation Classes and Swing Components** **a)** Write a Java Swing program to create a student registration form using `JFrame`, `JPanel`, `JLabel`, `JTextField`, and `JButton`. **b)** Write a Java program to demonstrate event handling using `ActionListener` and adapter classes. 2. **Swing Layouts and Event Handling** **a)** Write a Java Swing program to demonstrate different layout managers (`FlowLayout`, `BorderLayout`, `GridLayout`). **b)** Write a Java program to handle button click events using the Delegation Event Model. 3. **JDBC Connection and Database Operations** **a)** Write a Java program to connect to a MySQL database using JDBC and display a successful connection message. **b)** Write a Java program to insert and retrieve records using `Statement` and `ResultSet`. 4. **PreparedStatement and ResultSetMetaData** **a)** Write a Java program to perform insert and update operations using `PreparedStatement`. **b)** Write a Java program to display database table information using `ResultSetMetaData`. 5. **Transactions and Exception Handling in JDBC** **a)** Write a Java program to demonstrate transaction management using commit and rollback operations. **b)** Write a Java program to handle database exceptions using `try-catch-finally` in JDBC.
	Text Books 1. Herbert Schildt, Java The Complete Reference, Eleventh Edition, McGraw-Hill Education, 2020 2. Bryan Basham, Kathy Sierra, Bert Bates, Head First Servlets and JSP, O'reilly (SPD), 2018 3. Ivan Bayross, Web Enabled Commercial Applications Development Using Java 2, BPB Publications 4. Java XML and JSON: Document Processing for Java SE by Jeff Friesen January 2019, Apress
11	**Reference Books** 1. E. Balagurusamy, Programming with Java- A Primer, Tata McGraw-Hill Education India, 2023 2. Programming in JAVA, 2nd Ed, Sachin Malhotra & Saurabh Choudhary, Oxford Press, 2018 3. Joe Wigglesworth and Paula McMillan, Java Programming: Advanced Topics, Thomson Course Technology (SPD)

	4. Eric Jendrock, Jennifer Ball, D Carson and others, The Java EE 5 Tutorial, Pearson Education 5. Java Parsing Collection XML JSON: Map List XML JSON Transform by Yang Hu, 2019	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments · Journal Submission : 10 Marks · Assignments: 10 Marks · Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination: 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)	

Vertical – VI

Title of Paper: Yoga and Wellness I – NUCC303

Sr. No.	Heading	Particulars
1	**Description the course :** **Including but Not limited to :**	India is growing rapidly as a global super-power. To face the challenges of the century and to keep up with the pace of the world, maintaining health is of prime importance. Giving thrust to healthy society, Physical Education, Sports, Health & fitness and Yoga are of great significance in today's world. The Government of India insists on Physical Fitness, Mental Health and Overall Development of Personality for every citizen. However, creating efficient and skilled human resource in the field of Physical Education, Sports and Yoga is identified as the need of the hour. Thus, the Governments of India and Government of Maharashtra have included Physical Education, Sports and Yoga as a key area under the NEP 2020.
2	**Vertical :**	Co-Curricular
3	**Type :**	Activity Based
4	**Credit**:	2 credits
5	**Hours Allotted :**	30 Hours
6	**Marks Allotted**:	50 Marks
7	**Course Objectives**: The students will be able to: • **To introduce students to the historical roots and philosophical foundations of Yoga,** with emphasis on Patanjali's Yoga Sutras. • **To develop awareness and practical skills in different forms of Yoga** by learning basic asanas and understanding their physical and mental health benefits.	
8	**Course Outcomes**: **CO1:** Understand the origin, philosophy, and importance of Yoga as described in Patanjali's Yoga Sutras. **CO2:** Identify various types of Yoga and perform basic asanas effectively, demonstrating their benefits on physical and mental well-being.	

9	**Module :** This module introduces the meaning, origin, and significance of Yoga with reference to Patanjali's Yoga Sutras. It explains various types of yoga such as Hatha, Ashtanga, and Vinyasa. Students learn basic asanas like Tadasana, Bhujangasana, Trikonasana, and Vajrasana. The module highlights yoga's benefits in improving both physical fitness and mental well-being.
10	**Reference Books:** • Iyengar, B.K.S. (2001). Light on Yoga. London: Thorsons. • Tiwari, O.P. (1996). Asana: Why and How? Lonavla: Kaivalyadhama Yoga Institute.
11	**Internal Continuous Assessment: 40% (20 Marks)** ❖ Yoga Report/Assignment
12	External Assessment: 60% : **(30 Marks)** • Scanned copy of the Yoga participation certificate. • A short report on Yogasana (handwritten or typed). *Note: Students participating in sports competitions conducted by University at State or National Level, students who have represented Mumbai University or College at Intercollegiate / Inter Zonal / West Zone Inter University / All India Inter University/ International tournament are exempt from submission of report.*

Letter Grades and Grade Points:

Semester GPA/ Program CGPA Semester/ Program	% of Marks	Alpha-Sign / Letter Grade Result		Grade Points
9.00-10.00	90.0-100	O	(Outstanding)	10
8.00-<9.00	80.0-<90.0	A+	(Excellent)	9
7.00-<8.00	70.0-<80.0	A	(Very Good)	8
6.00-<7.00	60.0-<70.0	B+	(Good)	7
5.50-<6.00	55.0-<60.0	B	(Above Average)	6
5.00-<5.50	50.0-<55.0	C	(Average)	5
4.00-<5.00	40.0-<50.0	P	(Pass)	4
Below 4.00	Below 40	F	(Fail)	0
Ab (Absent)	-	AB	(Absent)	0

Signature of BOS Chairman
Dr. Bhakti Chaudhari
BOS (Computer Science)

Signature of Faculty Member
Ms. Rhucha Patil
(Computer Science)

Signature of I/C Principal
Dr. Swiddle D'Cunha

Nirmala Memorial Foundation College of Commerce and Science

(Autonomous)

Re-accredited by NAAC with B++, ISO 9001-2015 Certified, Recognised under section 2(f) & 12(B) of the UGC Act 1956

Permanently Affiliated to the University of Mumbai,

Syllabus

Bachelor of Science in Computer Science

(B.Sc. Computer Science)

(Programme Code: UGCS02)

As Per New Education Policy (NEP) 2020

(To be implemented from the Academic Year 2026-2027)

Approved in the Academic Council Meeting held on 8th ***June 2026***

Semester – IV

SEMESTER IV

COURSE STRUCTURE

VERTICAL		COURSE CODE	SUBJECT	CREDIT
Vertical 1	Major	NUCS401	Computer Networks	**2**
	Major	**NUCS402**	**Agile Software Development and DevOps**	**2**
	Major	NUCS403	IoT Technologies	**2**
	Major	NUCS404	Computer Science Practical 4	**2**
Vertical 2	Minor	NUCS405	DS_Advanaced Python for Data Science	2
	Minor	NUCS406	DS_Data Analysis with SAS/SPAA/R	2
Vertical 3	OE	NUCS407	Digital Payment Systems and Security	2
Vertical 4	SEC	NUCS408	Mobile Application Development	2
Vertical 5	AEC	NUEN402	Hindi Bhasha : Kaushal Ke Aadhar	**2**
Vertical 6	CC	NUCC403	Co-curricular	2
	CEP	NUCP401	Community Engagement Program	**2**
			Total Credits	**22**

Vertical – I

Mandatory Courses

Name of the Course: Computer Networks - NUCS401

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Computer Networks course provides a foundational understanding of how computers and other devices communicate over various types of networks. It introduces key networking models like OSI and TCP/IP, explores the structure and functionality of different layers such as physical, data link, network, transport, and application, and dives into real-world protocols such as Ethernet, IP, TCP, HTTP, and DNS. The course also addresses modern developments like IPv6, Quality of Service (QoS), and secure communication practices. **Relevance:** An era where connectivity drives innovation, understanding computer networks is essential for any computer science graduate. This course aligns with the growing need for professionals who can design, troubleshoot, and manage complex networked systems, including those used in cloud computing, data centers, mobile communications, and IoT environments. **Usefulness:** The course is highly useful for learners as it equips them with the theoretical background and practical insights needed to understand how data moves through networks. This knowledge is critical not only for roles in networking but also in software development, cybersecurity, systems administration, and IT infrastructure management. **Application:** Concepts learned in this course have direct application in configuring LANs and WANs, managing IP addresses, analyzing packet data, securing communications, and developing network-based applications. Students will be able to apply their knowledge in creating efficient, scalable, and secure communication systems. **Interest:** Students often find this course interesting due to its hands- on nature and immediate relevance to everyday technologies such as the internet, mobile phones, social media, and streaming services. Simulations and practical case studies help visualize how theoretical concepts apply in real-world network scenarios. **Connection with Other Courses:** This course connects closely with subjects such as Operating Systems, Database Management Systems, Web Programming, Cloud Computing, and Cybersecurity. A good understanding of networking is essential for understanding how distributed systems work, how servers handle requests, and how secure communication is maintained. **Demand in the Industry:** There is a consistent demand in the industry for professionals who

		understand networking principles and can manage network operations. Roles involving cloud platforms (like AWS, Azure), DevOps, and IT support all value networking skills. Knowledge of protocols and architectures is especially important for careers in network engineering, cybersecurity, and system integration. **Job Prospects:** Completing the Computer Networks course equips students for roles like network administrator, support technician, and system engineer. It builds a strong foundation for careers in cybersecurity, cloud computing, and IT infrastructure. The knowledge gained is valuable for industry certifications and in-demand across tech- driven sectors.
2	**Vertical:**	**Major**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Introduce fundamental concepts of computer networks, including types, architecture, and models like OSI and TCP/IP. 2. Explain data transmission techniques, switching methods, and network media. 3. Provide understanding of data link and network layer functions such as error handling, MAC, IP addressing, and routing. 4. Familiarize students with transport and application layer protocols, modern networking trends, and secure communication methods.	
8	**Course Outcomes:** After successful completion of this course, students would be able to – **CO 1.** Describe network models, architectures, and layered communication principles. **CO 2.** Analyze physical and data link layer functions including transmission media, error control, and MAC protocols. **CO 3.** Explain IP addressing (IPv4/IPv6), routing algorithms, and switching techniques. **CO 4.** Understand transport and application layer protocols and evaluate modern network challenges including QoS and security.	
9	**Modules:-** **Module 1 (15 hours):**	
	Introduction to Computer Networks: Networking standards and organizations (ISO, IEEE, IETF), Types of Networks: LAN, MAN, WAN, Network topologies and basic hardware **Network Models:** OSI Reference Model – layers, functions, TCP/IP Protocol Suite – layers and comparison with OSI **Physical Layer Concepts:** Data and signals: analog & digital, Signal impairments: attenuation, noise, distortion, Data transmission: bandwidth, throughput, latency, Digital transmission: line coding, analog-to-digital conversion, Transmission modes: simplex,	

	half-duplex, full-duplex **Bandwidth Utilization & Transmission Media:** Multiplexing: FDM, TDM, WDM, Spread Spectrum techniques: DSSS, FHSS, Transmission Media: Guided (Twisted Pair, Coaxial, Fiber Optics) & Unguided Media (Radio, Microwave, Infrared) **Switching Techniques:** Circuit Switching, Packet Switching (connectionless and connection-oriented) **Data Link Layer and Error Handling:** Link layer addressing (MAC), framing concepts, Error detection: Parity, CRC, Checksum, Error correction: Hamming Code, Data link protocols: Stop-and-Wait, Go-Back-N, HDLC, Introduction to MAC: CSMA/CD, CSMA/CA **Wired & Wireless LANs:** Ethernet (standard, fast, gigabit). IEEE 802.11 Wi-Fi, Bluetooth, WiMAX, Cellular telephony: Generations overview (2G–5G), Satellite networks: types and applications	
	Module 2 (15 hours):	
	Network Layer Fundamentals: Packet Switching Concepts, IPv4 Addressing, Subnetting, IP Packet forwarding & routing, Overview of ICMPv4, Basics of Mobile IP **Routing Techniques:** Routing algorithms: Concepts of Distance Vector & Link State Routing, Unicast Routing Protocols: Basic overview of RIP & OSPF **IPv6:** IPv6 Addressing format, comparison with IPv4, ICMPv6, Transition strategies **Transport Layer Protocols,** Transport layer services, User Datagram Protocol (UDP), Transmission Control Protocol (TCP): Features, 3-way handshake **Application Layer & Protocols:** Client-Server architecture: Iterative vs Concurrent Servers, Standard Protocols: HTTP (with HTTPS), FTP, Email (SMTP, POP3, IMAP), TELNET, SSH (modern replacement), DNS **Quality of Service (QoS):** Concepts of Delay, Jitter, Bandwidth, Flow control techniques, Integrated Services (IntServ) and Differentiated Services (DiffServ), Current trends: QoS in video streaming and VoIP	
10	**Text Books** 1. Data Communications and Networking, Behrouz A. Forouzan, Fifth Edition, TMH, 2018. 2. Computer Network, Andrew S. Tanenbaum, David J. Wetherall, Fifth Edition, Pearson Education, 2018.	
11	**Reference Books** 1. Computer Network, Bhushan Trivedi, Oxford University Press, 2016 2. Data and Computer Communication, William Stallings, PHI, 2017	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Class test of 15 marks Quizzes/ Presentations/ Assignments: 5 marks Total: 20 marks	**Format of Question Paper: External Examination (30 Marks)– 1 hr duration**

14	**Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)**

Questions	**Based On**	**Options**	**Marks**
Q.1	Module 1	Any 2 out of 4	10
Q.2	Module 2	Any 2 out of 4	10
Q.3	Module 1 & 2	Any 2 out of 4	10

Name of the Course: Agile Software Development & DevOps

Sr. No.	Heading	Particulars
1	**Description the course:**	This course introduces modern software development approaches focusing on **Agile methodologies** and the **DevOps culture**. It emphasizes the principles, models, and frameworks used to achieve iterative development, continuous delivery, and faster software deployment cycles. Students will gain knowledge of Agile project management, collaborative development practices, and the DevOps lifecycle that integrates development and operations for reliable software delivery.
2	**Vertical:**	**Major**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	Course Objectives ● To understand the philosophy and principles of Agile software development. ● To study various Agile methodologies and their applications in real-world projects. ● To introduce DevOps practices, culture, and lifecycle for faster software delivery. ● To familiarize students with configuration management, containerization, and orchestration concepts..	
8	Course Outcomes On completion of the course, students will be able to: 1. Explain the Agile Manifesto, principles, and different Agile methodologies. 2. Describe the roles, artifacts, and events of Scrum and compare Agile with traditional approaches. 3. Interpret Agile planning methods such as user stories, backlogs, and estimation techniques. 4. Understand the DevOps culture, lifecycle, and key practices for continuous delivery. 5. Explain the concepts of configuration management, containerization, and orchestration.	
9	**Modules: -** **Module 1 (15 hours):**	
	Module 1: Agile Software Development (15 hours) ● Introduction to Agile: Agile Manifesto, Principles, Agile vs. Traditional Models	

- Agile Methodologies: Scrum, Extreme Programming (XP), Kanban, Lean Development
- Agile Roles, Events, and Artifacts
- Requirements in Agile: User Stories, Backlogs, Estimation Techniques
- Agile Planning: Release Planning, Sprint Planning, Velocity
- Agile Practices: Continuous Integration, Continuous Delivery, Test-Driven Development, Pair Programming, Retrospectives
- Agile Metrics and Progress Tracking

Module 2 (15 hours):

Module 2: DevOps Fundamentals (15 hours)

- Introduction to DevOps: Definition, Principles, DevOps Culture and Lifecycle
- DevOps vs. Agile: Complementary Approaches
- DevOps Practices: Continuous Development, Continuous Integration, Continuous Testing, Continuous Deployment, Continuous Monitoring
- Configuration Management: Infrastructure as Code (IaC)
- Containerization: Concepts of Docker and Virtualization vs. Containerization
- Orchestration: Introduction to Kubernetes and its Components
- Monitoring and Logging in DevOps: Importance, Tools, Feedback Loops
- DevOps Tools Overview: Jenkins, GitHub Actions, Ansible, Docker, Kubernetes, Prometheus, Grafana

11 Reference Books

- *Agile Software Development: Principles, Patterns, and Practices* – Robert C. Martin
- *Essential Scrum: A Practical Guide to the Most Popular Agile Process* – Kenneth S. Rubin
- *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation* – Jez Humble & David Farley
- *The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win* – Gene Kim, Kevin Behr, George Spafford
- *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations* – Gene Kim, Jez Humble, Patrick Debois, John Willis

12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Class test of 15 marks Quizzes/ Presentations/ Assignments: 5 marks Total: 20 marks	**Format of Question Paper: External Examination (30 Marks)– 1 hr duration**

14 Format of Question Paper: (Semester End Examination: 30 Marks. Duration: 1 hour)

Questions	Based On	Options	Marks
Q.1	Module 1	Any 2 out of 4	10
Q.2	Module 2	Any 2 out of 4	10
Q.3	Module 1 & 2	Any 2 out of 4	10

Name of the Course: IoT Technologies - NUCS403

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Internet of Things (IoT) is an evolving field where devices connect and communicate to automate tasks and share data. This course lays the groundwork by explaining IoT's structure, history, key components, and its interaction with the internet and physical objects. **Relevance:** As industries shift toward automation and real-time analytics, understanding IoT has become vital for developing intelligent and connected systems. Its relevance spans domains like healthcare, agriculture, logistics, and smart cities. **Usefulness:** This course equips students with practical knowledge to create prototypes, interface sensors and actuators, and understand protocols. The hands-on exposure enhances the ability to design, implement, and troubleshoot IoT systems. **Application:** Students apply their knowledge in real-life scenarios such as environmental monitoring, smart homes, precision farming, and intelligent transportation systems. The course provides foundational skills to build IoT-based solutions from scratch. **Interest:** IoT offers an exciting blend of hardware, software, and communication, making it engaging for students who enjoy practical problem-solving. Working with microcontrollers and sensors adds a creative, hands-on dimension to learning. **Connection with Other Courses:** This course connects well with subjects like Embedded Systems, Computer Networks, Artificial Intelligence, Cloud Computing, and Mobile Application Development. It forms a practical bridge between theoretical concepts and real-world implementations. **Demand in the Industry:** The IoT market is booming with demand for professionals skilled in embedded programming, cloud integration, sensor interfacing, and data security. Industries are looking for people who can build, maintain, and scale IoT systems. **Job Prospects:** Students completing this course can explore roles like IoT Developer, Embedded Systems Engineer, Firmware Developer, IoT Solution Architect, and Application Developer. With further specialization, it opens doors to R&D, automation, and innovation labs in industries across the globe.

2	**Vertical:**	**Major**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** **1.** Introduce the fundamentals, architecture, and evolution of the Internet of Things (IoT). **2.** Explore IoT platforms, sensors, actuators, and communication protocols for device connectivity. **3.** Impart knowledge of system-on-chip (SoC) architecture and hardware interfacing techniques. **4.** Develop skills to design IoT applications integrated with web, cloud, and emerging technologies like Edge and Fog Computing.	
8	**Course Outcomes:** After successful completion of this course, students will be able to: **CO 1.** Explain the core concepts, architecture, and components of IoT systems. **CO 2.** Identify and interface sensors, actuators, and IoT boards like Arduino and Raspberry Pi. **CO 3.** Apply suitable communication protocols and ensure secure data transmission. **CO 4.** Design basic IoT applications with data acquisition, processing, and integration using cloud or edge technologies.	
9	**Modules:-** **Module 1 (15 hours):**	
	Introduction to IoT: Definition, Characteristics & Scope of IoT, History & Evolution of IoT, IoT vs M2M, IoT Architectures (Three-layer, Five-layer), Physical & Logical Design of IoT Systems, Enabling Technologies in IoT: Cloud Computing, Big Data, AI, Embedded Systems **IoT Components & Frameworks:** Smart "Things" and their identifiers, Overview of IoT Frameworks (Amazon AWS IoT, Google Cloud IoT, Azure IoT Hub) **System on Chip (SoC):** What is SoC? Structure & Characteristics, SoC Elements: FPGA, GPU, APU, Compute Units, Introduction to ARM and atmega328 Architectures **IoT Hardware Platforms:** Overview and comparison: Raspberry Pi, Arduino, NodeMCU, IoT board capabilities, selection criteria for applications **Hardware Interfacing & Communication Protocols:** Basic components: LED, Button, Camera, Motor, 8×8 LED Grid, Communication protocols: PWM, UART, GPIO, I2C, SPI	
	Module 2 (15 hours):	
	Sensors & Actuators: Digital and Analog Sensors: Temperature, Humidity, Motion, Light, Gas, Ultrasonic, Interfacing Relay Switch, Servo Motor **IoT Protocols & Security:** Protocols: HTTP, MQTT, CoAP, XMPP, UPnP, Privacy and	

	Security Issues in IoT **Web & Cloud Integration in IoT:** Web server setup for IoT, Data exchange with IoT device, Node-RED basics, Introduction to Cloud Platforms for IoT **Wireless Sensor Networks (WSNs):** Basics, Architecture, Types, Role in IoT communication **Edge & Fog Computing:** Definition, Purpose and Use Cases, Edge vs Fog vs Cloud comparison, Edge architectures and communication models, **IoT Applications:** Case Studies in Healthcare, Agriculture, Transportation, Smart Cities
10	**Text Books** 1. Introduction to IoT Paperback by Sudip Misra , Anandarup Mukherjee , Arijit Roy , Cambridge Press, 2022 2. Jain, Prof. Satish, Singh, Shashi, "Internet of Things and its Applications", 1st Edition, BPB, 2020. 3. Shriram K Vasudevan, Abhishek S Nagarajan, RMD Sundaram, Internet of Things, Wiley, India, 2019 4. IoT and Edge Computing for Architects - Second Edition, by Perry Lea, Publisher: Packt Publishing, 2020
11	**Reference Books** 1. Internet of Things by Vinayak Shinde, SYBGEN Learning India Pvt. Ltd, 2020 2. Internet of things, Dr. Kamlesh Lakhwani, Dr. Hemant kumar Gianey, Josef Kofi Wireko, Kamalkant Hiran, BPB Publication, 2020 3. Arduino, Raspberry Pi, NodeMCU Simple projects in easy way by Anbazhagan k and Ambika Parameswari k, 2019. 4. IoT based Projects: Realization with Raspberry Pi, NodeMCU Paperback – February 2020, by Rajesh Singh Anita Gehlot, 2020 5. Mastering the Raspberry Pi, Warren Gay, Apress, 2014
12	**Internal Continuous Assessment: 40%** / **Semester End Examination: 60%**
13	**Continuous Evaluation through:** Class test of 15 marks Quizzes/ Presentations/ Assignments: 5 marks Total: 20 marks **Format of Question Paper: External Examination (30 Marks)– 1 hr duration**
14	**Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)**

Questions	Based On	Options	Marks
Q.1	Module 1	Any 2 out of 4	10
Q.2	Module 2	Any 2 out of 4	10
Q.3	Module 1 & 2	Any 2 out of 4	10

Name of the Course: Computer Science Practical 4 – NUCS404

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** This major practical course is a blend of two rapidly evolving areas in computing — Computer Networking and the Internet of Things (IoT). While the first part equips students with hands-on experience in network creation, routing, and traffic analysis, the second part transitions them into the world of intelligent connected devices. Students explore the design and deployment of smart systems using Raspberry Pi and Arduino kits, integrate sensors/actuators, and implement cloud-based interactions. **Relevance:** This course is highly relevant in today's digital world where interconnected devices and reliable communication networks form the backbone of every industry. The integration of Computer Networks and IoT ensures students gain comprehensive knowledge about both data communication and real-time embedded systems. As businesses and governments adopt automation and smart systems, professionals with expertise in both domains are in increasing demand. **Usefulness:** The course equips students with practical skills to configure networks, interface hardware, and use protocols for communication and control. It bridges the gap between theory and practice by allowing learners to build, simulate, and test complete systems. These hands on experiences enhance understanding and build a strong foundation for advanced learning or employment. **Application:** The knowledge gained in this course can be directly applied to developing IoT systems like smart homes, environmental monitoring, health devices, and automated machinery. Students also learn to simulate network infrastructures and deploy routing protocols, which are essential in setting up enterprise-level networks. Real- world applications of this course span domains such as agriculture, transportation, education, and urban planning. Interest: This course captures students' interest through interactive hardware activities like blinking LEDs, controlling motors, and streaming sensor data. Tools like Cisco Packet Tracer, Node-RED, and Wireshark provide engaging visual and analytical insights into networking and IoT systems. Students are encouraged to innovate and experiment, keeping the learning experience dynamic and enjoyable. **Connection with Other Courses:** The course complements core subjects such as Computer Networks, Microprocessor and Embedded Systems, Operating Systems, and Web Programming. It allows students to apply concepts from these theory courses in a practical setup, promoting interdisciplinary learning. The use of programming and cloud communication also ties into courses like Data Science and Cloud Computing. **Demand in the Industry:** There is a significant and growing demand for professionals skilled in IoT and network technologies across industries such as healthcare, smart infrastructure, manufacturing, and IT services. Companies look for

		candidates who can develop, deploy, and maintain systems involving both physical hardware and software networking components. This course provides the skillset that aligns with industry trends and emerging technologies. **Job Prospects:** Students completing this course are better prepared for roles such as IoT Developer, Network Administrator, Embedded Systems Programmer, and Cloud-IoT Integrator. They can work in sectors including telecom, automation, R&D, and smart technologies. The practical skills gained make them suitable for both core technical roles and interdisciplinary project teams.
2	**Vertical:**	**Major**
3	**Type:**	**Practical**
4	**Credits:**	**2 credits(1 credit = 30 Hours of Practical work in a semester)**
5	**Hours Allotted:**	**60 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** **1.** Develop practical skills in computer networking using tools like Cisco Packet Tracer, including network setup and configuration. **2.** Explore IP addressing, routing protocols (RIP, OSPF, BGP), and network topologies. **3.** Understand IoT communication protocols and interface sensors and actuators using platforms like Raspberry Pi and Arduino. **4.** Build integrated IoT applications using tools like Node-RED and cloud platforms for data visualization and control.	
8	**Course Outcomes:** After successful completion of this course, students will be able to: **CO 1.** Use system commands and simulation tools to configure and troubleshoot wired/wireless networks. **CO 2.** Design and analyze networks using routing protocols and traffic monitoring tools like Cisco Packet Tracer and Wireshark. **CO 3.** Configure IoT hardware and interfaces for real-time device communication and data acquisition. **CO 4.** Develop integrated IoT solutions using sensors, cloud platforms, and web-based dashboards.	
9	**Modules:-** **Module 1 : Practical based on Computer Networks (30 hours):**	
	Exploring Networking Commands via Windows CMD / LINUX Terminal • Execute and observe the output of: ping, traceroute / tracert, netstat, arp, ipconfig / ifconfig, getmac, hostname, nslookup, pathping, systeminfo • Discuss the purpose and interpretation of each command's output **Learning Focus:** Understanding diagnostic and configuration commands. **Building a Basic Peer-to-Peer Network** Create a network with two PCs connected using a crossover cable using Cisco Packet Tracer • Assign static IP addresses • Test connectivity with ping and Packet Tracer simulation	

Static IP Setup with One Server and Two Clients

Connect **1 server** and **2 computers** using a switch (Cisco Packet Tracer)

- Use **static IP addresses**
- Configure server services (e.g., HTTP or FTP) and test connectivity from clients

Dynamic IP Allocation with Server and Clients

Use **DHCP service** from a server to assign IPs to **two PCs** (Cisco Packet Tracer)

- Enable and configure DHCP on the server
- Verify IP allocation and connectivity using ipconfig

Creating a Mixed Network with Wired and Wireless Devices

One server, two wired PCs, and two **mobile/wireless devices (**Cisco Packet Tracer**)**

- Use appropriate cabling and access points
- Assign IPs and test cross-device communication

RIP Version 1 Routing Across Three Routers

Three routers, each connected to at least **three PCs (**Cisco Packet Tracer**)**

- Implement **RIPv1** routing between routers
- Verify inter-network connectivity using ping and route tables

RIP Version 2 Implementation

Three routers, each connected to at least **three PCs (**Cisco Packet Tracer**)**

- Enable **RIPv2** and observe subnet mask handling
- Use Packet Tracer's simulation mode to observe routing updates

OSPF Routing and Network Hierarchies

Three routers and their networks using **OSPF**

(Three routers, each connected to at least **three PCs (**Cisco Packet Tracer**)**

- Assign area IDs, router IDs, and enable OSPF
- Monitor OSPF neighbour relationships and path selections

BGP for Inter-domain Routing

Three autonomous systems (AS) with routers running **BGP** Three routers, each connected to at least **three PCs (**Cisco Packet Tracer**)**

- Configure BGP with different AS numbers
- Establish peerings and test inter-AS communication

Protocol Analysis with Wireshark

- Set up network transactions for each protocol:
  ICMP (ping), TCP (web browsing), HTTP (via browser), UDP (DNS), FTP (file transfer)

• Apply filters and observe packet contents

Module 2: Practical based on IoT Technologies (30 hours)

Preparing the IoT Hardware

· Set up Raspberry Pi OS / Arduino IDE

· Configure GPIO settings and test basic connectivity

· Demonstrate pin layout and onboard peripherals

GPIO – Light the LED (with and without Button)

· Blink LED using Python (Raspberry Pi) or C++ (Arduino)

· Add a push button to toggle LED ON/OFF

SPI Interface – Camera Module Integration

· Connect a Pi camera module (or SPI camera for Arduino)

· Capture an image or short video

	· Store file or stream it locally **8x8 LED Grid Control (Matrix LED Programming)** · Connect an 8×8 LED matrix module · Program animations or scrolling text patterns · Explore logical formulas for patterns **PWM – Stepper Motor Control** · Interface a stepper motor using a motor driver · Control direction and vary speed using PWM signals · Observe effect of duty cycle changes on motor movement **Node-RED for IoT Dashboard** · Install and configure Node-RED on Raspberry Pi · Create a flow to turn LED ON/OFF via browser · Add visual interface for sensor data (e.g., temperature) **Sensor Integration – Analog & Digital Sensors** · Interface multiple sensors (LDR, DHT11, Gas) · Collect and display data on serial monitor / OLED / LCD · Trigger actions (e.g., fan ON if temp > threshold) **Web Trigger – Control GPIO from Web Server** · Host a simple Flask web app (Raspberry Pi) or ESP Web Page (Arduino) · Control a set of LEDs via buttons on a webpage · Ensure real-time response and feedback **IoT Protocol – Send Sensor Data Online** · Use HTTP or MQTT to push sensor values to a cloud server (e.g., Thingspeak) · Implement publishing logic with timestamps · Use Arduino/NodeMCU or Raspberry Pi as the publisher **Integration – Smart Monitoring System** · Combine sensors, actuators, communication protocol, and web/cloud · Example: A Smart Weather Station that logs temp/humidity online and triggers fan/LED alerts ***Note: The above practicals can be performed on Raspberry Pi Kits and / or Arduino kits as per the need***	
10	**Text Books** 1. Kurose, J.F. & Ross, K.W. (2021). *Computer Networking: A Top-Down Approach* (7th ed.). Pearson Education India. 2. Ramya, V., & Shanmuga Priya, K. (2019). *Practical Internet of Things: Concepts, applications and security*. Chennai: Wiley India Pvt. Ltd.	
11	**Reference Books** 1. Forouzan, B.A. (2017). *Data Communications and Networking* (5th ed.). McGraw Hill Education India. 2. Bahga, A. & Madisetti, V. (2014). *Internet of Things: A Hands-on Approach*. Universities Press India.	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments • Journal Submission : 10 Marks • Assignments: 10 Marks Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*

14	Format of Question Paper: (Semester End Practical Examination : 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)

Vertical – II

Minor Courses

Name of the Course: Advance Python for Data Science – NUCS405

Sr. No.	Heading	Particulars
1	**Description the course:**	Advanced Python for Data Science is designed to equip learners with essential data manipulation, analysis, and visualization skills using Python's powerful libraries – Pandas, NumPy, and Seaborn. The course focuses on efficiently handling large datasets, performing complex data transformations, statistical analysis, and creating insightful visualizations. Learners will gain practical experience through hands-on labs, enabling them to apply these techniques to real-world data science problems.
2	**Vertical:**	**Minor**
3	**Type:**	**Practical**
4	**Credits:**	**2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. To introduce students to data manipulation and analysis using Pandas and NumPy, including data loading, transformation, and filtering techniques. 2. To equip students with the ability to perform statistical computations, matrix operations, and time-series analysis using Pandas and NumPy. 3. To develop proficiency in data visualization using Seaborn, enabling students to create meaningful visual representations of datasets. 4. To train students in using Seaborn for advanced plotting techniques, including heatmaps, pairplots, and time-series visualizations for data-driven insights.	
8	**Course Outcomes:** After successful completion of this course, students would be able to - CO1. Students will be able to manipulate and preprocess datasets using Pandas and NumPy, including handling missing values, filtering data, and performing statistical analysis. CO2. Students will perform matrix operations, generate random datasets, execute Boolean indexing, and apply time-series analysis using NumPy and Pandas. CO3. Students will create and customize various Seaborn plots, including histograms, boxplots, scatter plots, and violin plots for effective data visualization. CO4. Students will apply advanced Seaborn visualization techniques such as heatmaps, facet grids, and time-series plots to extract and present meaningful insights from data.	
9	**Modules:-** **Module 1 (15 hours):**	
	Practical 1: Data Loading, Inspection, and Cleaning with Pandas · Load a CSV dataset into a Pandas DataFrame.	

	· Display the first few rows, check data types. · Handle missing values, remove duplicates. · Rename columns, change data types, drop unnecessary columns. **Practical 2:** Advanced Data Manipulation with Pandas · Add a new calculated column. · Filter rows using .loc and .query() (e.g., sales above a threshold, region = "North") · Merge two DataFrames using inner, outer, left, right joins (e.g., customers + orders). **Practical 3:** Statistical Analysis with NumPy · Generate a random dataset of 1000 values (or a 1D array of 100 random integers between 1 and 1000). · Calculate mean, median, variance, standard deviation, min, max. · Apply Boolean indexing (filter values > 500 and < 800, compute mean of filtered). **Practical 4:** Matrix Operations with NumPy · Create two 3x3 matrices of random integers. · Perform matrix addition, subtraction, multiplication, element-wise division. · Calculate determinant and inverse of a matrix. **Practical 5:** Grouping, Pivoting, and Time Series Analysis with Pandas · Group dataset by one or more categorical columns and calculate count, mean, std. · Create a pivot table showing product-wise sales for each region. Perform time series analysis: extract year, month, day, weekday; group data by month, calculate monthly sales. ***Note : Create a mini assignment for students that covers the above concepts in a single question***
	Module 2 (15 hours):
	Practical 6: Histogram, Boxplot, and Barplot Visualization with Seaborn · Load a dataset into Pandas. · Plot a histogram showing the distribution of a numerical column. Customize bin size, color, and title. · Plot a boxplot of a numerical column grouped by a categorical column (e.g., salary by department). · Create a barplot comparing average values of a numerical column for categories of a categorical column. Customize axes labels and title. **Practical 7:** Pairwise and Faceted Visualizations with Seaborn · Create a Seaborn pairplot to visualize pairwise relationships between numerical columns. Add hue for categories. · Create a Seaborn facet grid of scatter plots showing relationships between two numerical columns for different values of a third categorical column. **Practical 8:** Correlation Heatmap and Violin Plot · Generate a correlation matrix from a DataFrame and create a Seaborn heatmap. Customize color palette, annotations, and title. · Create a violin plot showing the distribution of multiple numerical columns grouped by a categorical column. **Practical 9:** Scatter Plot and Count Plot · Create a Seaborn scatter plot between two numerical columns, add hue to differentiate categories, customize markers and plot size. · Create a count plot for a categorical column, customize colors, orientation, and add value labels. **Practical 10:** **Concept:** Line Plot for Time Series · Create a Seaborn line plot to visualize trends in a time series dataset. · Customize with appropriate labels, grid lines, and title.

	Note : Create a mini assignment for students that covers the above concepts in a single question	
10	**Text Books** 1. Python for Data Analysis by Wes McKinney 2nd Edition Publisher: O'Reilly Media	
11	**Reference Books** 1. Python Data Science Handbook by Jake VanderPlas 1st Edition Publisher: O'Reilly Media 2. Hands-On Data Analysis with Pandas by Stefanie Molin 2nd Edition Publisher: Packt Publishing	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments • Journal Submission : 10 Marks • Assignments: 10 Marks Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination : 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)	

Name of the Course: Data Analysis with SAS / SPSS /R – NUCS406

Sr. No.	Heading	Particulars
1	**Description the course:**	Data Analysis with SAS / SPSS /R course provides hands-on training in data analysis techniques using industry-standard tools — SAS, SPSS, or R. It covers essential skills such as data importing, cleaning, transformation, and visualization, along with performing statistical analysis like t-tests, ANOVA, chi-square tests, and regression. Learners will gain practical experience in managing datasets, applying statistical methods, and generating professional reports, preparing them for real-world data analysis tasks across various domains.
2	**Vertical:**	**Minor**
3	**Type:**	**Practical**
4	**Credits:**	**2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. To understand the SAS/SPSS/R as a tool to data analysis. 2. Understand different techniques of data manipulation. 3. Use different functions for descriptive statistics. 4. To generate the reports after data manipulation.	
8	**Course Outcomes:** After successful completion of this course, students would be able to - CO1. To understand the basis of data analytics using a software tool. CO2. To use data from multiple sources relevant for Data Analytics. CO3. To Categorize and utilize data for inferential data analytics. CO4. To integrate the statistical tool with an analytical problem to bring the proficiency.	
9	**Modules:-** **Module 1 (15 hours):**	
	Data Handling, Preparation and Transformation (using SAS or SPSS or R) 1. Introduction to the SAS, SPSS and R environments – installation, interface overview, and loading data files. 2. Creating datasets from raw data (text files, CSV files, Excel sheets) and importing data into SAS/SPSS/R. 3. Exploring data: Displaying datasets using PROC PRINT (SAS), Data Viewer (SPSS), and View() or print() (R). 4. Applying conditional filters using IF, WHERE, and IF-THEN in SAS;Select Cases in SPSS; and subset() or filter() in R.	

5. Sorting data using PROC SORT in SAS, Sort Cases in SPSS, and arrange() in R.
6. Combining and appending datasets using MERGE in SAS, Merge Files in SPSS, and merge() or bind_rows() in R.
7. Selecting and dropping variables using KEEP, DROP in SAS,Variable View in SPSS, and select() in R.
8. Applying basic data cleaning functions: handling missing values using MISSING (SAS), Define Missing Values (SPSS), and na.omit()/replace_na() in R.
9. Performing text manipulation using substr, scan (SAS), String functions (SPSS), and str_sub(), str_split() (R).
10. Creating new variables using transformations and calculations (all three software).
11. Reshaping data using PROC TRANSPOSE (SAS), Restructure Data Wizard (SPSS), and pivot_longer()/pivot_wider() (R).
12. Combining datasets vertically (concatenation) using SET statement (SAS), Merge Files - Add Cases (SPSS), and rbind() (R).
13. Identifying and handling duplicates using PROC SORT NODUPKEY (SAS), Identify Duplicate Cases (SPSS), and distinct() (R).
14. Extracting date components using DATE functions (SAS), Date & Time Wizard (SPSS), and lubridate:: functions (R).
15. Generating basic summaries using PROC CONTENTS (SAS), Variable View (SPSS), and str() or summary() (R).

Module 2 (15 hours):

Statistical Analysis and Reporting (using SAS or SPSS or R)

1. Generating descriptive statistics using PROC MEANS (SAS), Descriptive Statistics (SPSS), and summary() or describe() (R).
2. Generating frequency tables using PROC FREQ (SAS), Frequencies (SPSS), and table() or count() (R).
3. Creating cross-tabulations and two-way tables using PROC FREQ (SAS), Crosstabs (SPSS), and table() (R).
4. Performing one-sample t-tests using PROC TTEST (SAS), T-Test (SPSS), and t.test() (R).
5. Performing independent two-sample t-tests using PROC TTEST (SAS), T-Test (SPSS), and t.test() with grouping (R).
6. Performing paired t-tests using PROC TTEST (SAS), Paired Samples T-Test (SPSS), and t.test(paired=TRUE) (R).
7. Performing one-way ANOVA using PROC ANOVA (SAS), One-Way ANOVA (SPSS), and aov() (R).
8. Performing two-way ANOVA using PROC GLM (SAS), Univariate Analysis (SPSS), and aov() (R).
9. Conducting Chi-square tests using PROC FREQ (SAS), Crosstabs with Chi Square (SPSS), and chisq.test() (R).
10. Creating graphical reports using PROC REPORT (SAS), Chart Builder (SPSS), and ggplot2 (R).
11. Generating histograms and box plots using PROC SGPLOT (SAS), Graphs (SPSS), and

	ggplot2 (R). 12. Generating correlation matrices using PROC CORR (SAS), Correlation (SPSS), and cor() (R). 13. Performing linear regression analysis using PROC REG (SAS), Regression (SPSS), and lm() (R). 14. Performing logistic regression using PROC LOGISTIC (SAS), Binary Logistic Regression (SPSS), and glm() (R). 15. Exporting results into external files (Excel, CSV,PDF) using ODS (SAS), Export Wizard (SPSS), and write.csv()/writexl (R). Note : Create a mini assignment for students that covers the above concepts in a single question	
10	**Text Books** 1. The Little SAS Book: A Primer – Lora D. Delwiche & Susan J. Slaughter 2. Learning SAS by Example: A Programmer's Guide – Ron Cody 3. Discovering Statistics Using IBM SPSS Statistics – Andy Field 4. IBM SPSS for Introductory Statistics: Use and Interpretation – George A. Morgan, Nancy L. Leech, 5. R for Data Science – Hadley Wickham & Garrett Grolemund 6. The Book of R: A First Course in Programming and Statistics – Tilman M. Davies	
11	**Reference Books** 1. Applied Statistics and the SAS Programming Language – Ron P. Cody & Jeffrey K. Smith 2. Data Analysis with IBM SPSS Statistics – Kenneth Stehlik-Barry & Anthony J. Babinec 3. Hands-On Programming with R – Garrett Grolemund	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments • Journal Submission : 10 Marks • Assignments: 10 Marks Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination : 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)	

Vertical – III

Name of the Course: Digital Payment Systems and Security

Sr. No.	Heading	Particulars
1	**Description the course:**	This course provides insights into digital payment systems, methods, and security frameworks. It covers electronic fund transfer systems, mobile wallets, UPI, card payments, and e-commerce transactions, while emphasizing fraud detection, data privacy, and regulatory standards. Students will gain a practical understanding of how digital payments operate in India and globally, along with challenges in ensuring security and trust.
2	**Vertical:**	**Open Electives**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**
7	**Course Objectives:** 1. Understand the evolution and types of digital payment systems. 2. Familiarize with Indian and global digital payment infrastructure. 3. Learn security measures like encryption, authentication, and fraud prevention. 4. Explore regulatory and policy frameworks for digital payments. 5. Appreciate future innovations and trends in payment technologies.	
8	**Course Outcomes:** **After completing this course, students will have the knowledge and skills to** CO1: Describe the different digital payment methods and platforms. CO2: Explain UPI, wallets, card systems, and e-commerce transactions. CO3: Evaluate risks and security measures in digital payments. CO4: Understand RBI, NPCI, and international regulations in payment systems. CO5: Analyze emerging trends like contactless and biometric payments.	
	Modules:- Module 1 : Digital Payment Systems in India and Abroad (15 hours) • Introduction to digital payments – meaning, advantages, limitations • Evolution of payments – from cash to digital • Indian systems: UPI, IMPS, NEFT, RTGS, AePS (Aadhaar Enabled Payment System) • Mobile wallets (Paytm, PhonePe, Google Pay) and prepaid cards • Debit cards, credit cards, and Point-of-Sale (POS) terminals • E-commerce payment systems – Cash on Delivery (CoD), net banking, card payments • International payment platforms: PayPal, Apple Pay, Google Pay (global), Stripe	

Module 2 : Security, Regulation, and Future Trends (15 hours)

- Importance of security in digital payments
- Security measures: encryption, OTP, biometric authentication, tokenization
- Common cyber threats – phishing, malware, frauds in digital payments
- Role of NPCI (National Payments Corporation of India) and RBI regulations
- PCI-DSS compliance and data privacy rules
- Fraud detection tools and real-world examples of payment frauds
- Future trends: contactless payments, biometric payments, blockchain in payments

10 Text Books

1. R.K. Sinha. Digital Payment Systems.
2. NPCI & RBI official guidelines (latest versions).

11 Reference Books

1. Dave Birch. Identity is the New Money.
2. Bhaskar, R. Electronic Payment Systems: Security and Legal Issues.

12 **Internal Continuous Assessment: 40%** | **Semester End Examination : 60%**

13 **Continuous Evaluation through:**

Sr. No	Particular	Marks
1.	Class Test	15
2.	Class participation	05
	Total	**20**

Semester End Examination (30 Marks)

14 Format of Question Paper: (Semester End Examination: 30 Marks. Duration:1 hour)

Questions	Type & Module	Marks
Q1.	Objective Question A) Multiple choice questions B) True or False	 08 Marks 07 Marks
Q2.	Theory Question	15 Marks
	OR	
Q2.	Theory Question	15 Marks
	Total	**30**

Vertical – IV

Vocational & Skill Enhancement Courses (VSEC)

Name of the Course: Mobile Application Development – NUCS408

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** Mobile devices are ubiquitous in modern life, and with them comes the need for innovative and functional mobile applications. This course introduces students to the world of Android app development using Kotlin, the officially recommended programming language by Google. Learners get hands-on exposure to building real-time applications using Android Studio, enabling them to create robust, secure, and user-friendly mobile applications. **Relevance:** This course bridges academic programming knowledge with practical industry skills. In a world where mobile- first solutions dominate, Android development is one of the most sought-after technical skills, with Kotlin leading the trend due to its expressive syntax and reliability. **Usefulness:** This course provides hands-on experience with real-time Android app development using Kotlin, helping students' bridge theoretical knowledge and practical application. Learners gain exposure to essential skills like UI design, database integration, and deployment practices, which are critical in the modern software development cycle. **Application:** Students can build interactive apps for domains like education, health, business, and entertainment. Through the use of features like Firebase, media handling, and location services, learners can design complete, data- driven mobile solutions suitable for real-world implementation. **Interest:** The course is engaging due to its visual and interactive nature — students see their code come to life in the form of functioning apps. Working with media, animations, sensors, and camera access makes the learning process exciting and creatively fulfilling. **Connection with Other Courses:** This course builds directly on programming concepts taught in Object-Oriented Programming and integrates well with Database Management Systems through SQLite and Firebase. It also relates to Software Engineering principles for structured app development and UI/UX Design for front-end aesthetics. **Demand in the Industry:** Android dominates the global mobile market, making Android app development a high-demand skill. With Google's backing of Kotlin, there is a rising need for developers proficient in this language, especially in the mobile-first product ecosystem. **Job Prospects:** Successful completion opens doors to roles like Android Developer, Mobile App Engineer, UI/UX Developer, and Firebase Backend Integrator. Students can also work as freelancers, contribute to tartups, or launch their own apps on the Google Play Store.

2	**Vertical:**	VSC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours of Practical work in a semester)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives:** **1.** Understand the fundamentals of mobile app development using Kotlin and Android Studio. **2.** Design and build user-friendly and responsive Android interfaces using XML and Kotlin. **3.** Implement core Android components such as Activities, Intents, Fragments, and services. **4.** Manage data storage and cloud integration using SQLite and Firebase, and prepare apps for deployment and publishing	
8	**Course Outcome:** After successful completion of this course, students will be able to: **CO 1.** Develop interactive Android applications using Kotlin and Android Studio setup. **CO 2.** Implement modular app features using Activities, Intents, Fragments, and UI components. **CO 3.** Integrate local and cloud-based data storage using SQLite and Firebase. **CO 4.** Deploy mobile applications with multimedia, location services, and background processing.	
9	**Modules:-** **Module: (15 hours)**	
	Getting Started with Android Studio & Kotlin: Setting up Android Studio, AVD, and first Kotlin-based app, Kotlin basics: variables, data types, type conversion, operators, Simple user input/output using TextView, EditText, Button **Kotlin Control Flow & OOP Basics in Action:** Control statements: if, when, loops, Functions, default arguments, extension functions, OOP concepts: classes, objects, inheritance, companion object **Android Core Components:** Activities, Intents (explicit/implicit), Activity lifecycle: demo with logs and state changes, Fragments: modular UI creation, Toasts, Dialogs, and simple navigation **Layout Design & UI Interactions:** Layouts: Linear, Relative, ConstraintLayout, Views and UI Controls: TextView, EditText, Button, ImageView. Event handling: onClickListener, simple data validation, Styling UI: themes, styles, and manifest configuration	
	Module: (15 hours)	
	Working with Lists & Menus: ListView, RecyclerView, Adapter usage, Menus: options, context, and popup menus, Fragments with RecyclerView navigation **Data Persistence & Firebase Integration:** SharedPreferences, SQLite basics and CRUD operations, Firebase Realtime Database: read/write, rules, testing, Dynamic UI based on data **Multimedia, Animations & Camera Access:** Using ImageView, switching images, Playing audio using MediaPlayer, Simple animations with XML, Accessing device camera (capture & display) **Location, Background Tasks & App Deployment:** Accessing location (GPS),	

	Background tasks using JobScheduler, App signing and versioning, Deploying to Google Play (demo or mock submission)	
	Text Books 1. How to Build Android Apps with Kotlin: A hands-on guide to developing, testing, and publishing your first apps with Android, Alex Forrester, Packt Publishing, 2021 2. Android Programming: Crafting UI/UX using Kotlin, SYBGEN Learning, 2020	
11	**Reference Books** 1. Head First Android Development: A Learner's Guide to Building Android Apps with Kotlin Dawn Griffiths, 3rd Edition, O'Reilly Media, 2021 2. Android Studio 4.2 Development Essentials- Kotlin Edition: Developing Android Apps Using Android Studio 4.2, Kotlin and Android Jetpack, Neil Smyth, Payload Media, 2021 3. Android Programming with Kotlin for Beginners, John Horton, Packt Publishing, 2019	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	**Continuous Evaluation through:** Practical journal submission, viva, assignments Journal Submission : 10 Marks Assignments: 10 Marks Total: 20 marks	**A semester end practical examination of 2 hours duration for 30 marks as the paper pattern given below.** *Its compulsory to carry certified journal at the time of practical exam*
14	Format of Question Paper: (Semester End Practical Examination: 30 Marks. Duration:2 hours) Q1: Module 1 (12 marks) Q2: Module 2 (12 marks) Q.3 Viva (06 Marks)	

Vertical – V

Name of the Course: हिंदी भाषा : कौशल के आधार

Sr. No.	Heading	Particulars
1	**Description the course:**	हिन्दी राजभाषा होने के साथ-साथ भारत में बोली जाने वाली एक प्रमुख भाषा है। भारत के अधिकांश निवासी और यहां तक कि भारत के बाहर बसने वाले भारतवंशी भी अपने दैनिक आपसी वार्तालाप, कार्य-व्यवहार में हिन्दी भाषा का ही प्रयोग करते हैं। विश्व की प्रमुख पाँच भाषाओं के अंतर्गत हिन्दी का अस्तित्व है, इस दृष्टि से हिन्दी को लेकर विभिन्न प्रकार के कौशल सीखे और सिखाए जा सकते हैं। विद्यार्थियों के लिए हिन्दी एक सामान्य भाषा होने के साथ विशेष भाषा तब बन जाती है जब वह हिन्दी के माध्यम से अपने कौशल में अभिवृद्धि करें, हिन्दी के माध्यम से रोजगार के कई अवसरों को प्राप्त करें। इस दृष्टि से पाठ्यक्रम अत्यंत लाभवर्धक और उपयोगी सिद्ध होगा। हिन्दी भाषा में कौशल विकास की असीम संभावनाएँ हैं और कौशल के विभिन्न आयाम जुड़े हुए हैं जो अलग-अलग दिशाओं में देखे जा सकते हैं। पाठ्यक्रम विद्यार्थियों में लेखन, वाचन कौशल की अभिवृद्धि करने के साथ रोजगारपरक अवसर प्रदान करता है।
2	**Vertical:**	**AEC**
3	**Type:**	**Theory**
4	**Credits:**	**2 credits**
5	**Hours Allotted:**	**30 Hours**
6	**Marks Allotted:**	**50 Marks**

7	**Course Objectives:** 1. विद्यार्थियों को लेखन, वाचन कौशल का ज्ञान देना एवं रोजगार के अवसरों से जोड़ना। 2. विद्यार्थियों को लेखन, वाचन कौशल से परिचय कराते हुए अभिव्यक्ति की शैलियों का विकास करना। 3. विद्यार्थियों को भाषण कला के विविध रूपों को समझाना, मौलिकता में अभिवृद्धि लाना एवं विशेषज्ञता दिलाना। 4. विद्यार्थियों को श्रवण कौशल की विशेषताओं से परिचय कराते हुए श्रवण कौशल के लाभों से अवगत कराना।
8	**Course Outcomes:** **CO-1)** विद्यार्थियों को लेखन, वाचन कौशल के ज्ञान प्राप्ति के साथ मौलिक अभिव्यक्ति में बदलाव आएगा। **CO-2)** विद्यार्थियों का लेखन, वाचन कौशल द्वारा मानसिक विकास होगा, पठन-शक्ति, शैली का विकास होगा। **CO-3)** विद्यार्थियों को लेखन, भाषण कौशल से भाषिक-शक्ति, शैलियों का संवर्धन होगा, विशेषज्ञता आएगी। **CO-4)** विद्यार्थियों को लेखन, वाचन, श्रवण, भाषण कौशल की विशेषताओं और उपयोगिता का ज्ञान प्राप्त होगा।
9	**Modules:-**
	इकाई – 1 व्याख्यान – 15 | क्रेडिट – 01 • लेखन कौशल का अर्थ एवं स्वरूप • लेखन कौशल की उपयोगिता एवं महत्व • लेखन कौशल की विधियाँ • लेखन कौशल के भेद एवं विशेषताएँ • वाचन कौशल का अर्थ, स्वरूप एवं विशेषताएँ • वाचन कौशल की उपयोगिता • वाचन कौशल की विधियाँ एवं विशेषताएँ
	इकाई – 2 व्याख्यान – 15 | क्रेडिट – 01 • भाषण कौशल का अर्थ एवं स्वरूप • भाषण कौशल का महत्व एवं उपयोगिता • भाषण कौशल की विशेषताएँ • भाषण कौशल की विधियाँ • श्रवण कौशल का अर्थ एवं स्वरूप • श्रवण कौशल का महत्व एवं उपयोगिता • श्रवण कौशल की विशेषताएँ

10	**संदर्भ ग्रंथ** • **हिन्दी भाषा शिक्षण के विविध आयाम** – प्राध्यापक डॉ. राठौर, किनले एडिशन। • **अभिनव पत्र लेखन** – डॉ. अनिल सिंह। • **हिन्दी के व्यावहारिक रूप** – डॉ. संतोष मोटवानी, परिदृश्य प्रकाशन, मुंबई। • **हिन्दी भाषा लेखन कौशल** – गुलीबाबा पब्लिकेशन प्राइवेट लिमिटेड।	
11	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
	Continuous Evaluation through: **आंतरिक मूल्यांकन – 20 अंक** • रचनात्मक कार्य / प्रकल्प इत्यादि – **10 अंक** • कक्ष शिक्षण के दौरान सहभागिता इत्यादि – **05 अंक** • अकादमिक, व्यावसायिक एवं कौशल संवर्धन गतिविधियाँ – **05 अंक** **कुल : 20 अंक**	**बाह्य मूल्यांकन – लिखित परीक्षा – 30 अंक** **परीक्षा अवधि – 01 घंटा** **निम्नलिखित तीन में से किन्हीं दो प्रश्नों के उत्तर लिखिए। 30 अंक** **कुल योग – 30 अंक**
13	**Format of Question Paper:** ***बाह्य मूल्यांकन- लिखित परीक्षा- 30- अंक*** ***परीक्षा अवधि- 01 घंटा*** ***निम्नलिखित तीन में से किन्हीं दो प्रश्नों के उत्तर लिखिए*** ***30 अंक*** कुलयोग- **30 अंक**	

Vertical – VI

Name of the Course: Yoga and Wellness II – NUCP401

Sr. No.	Heading	Particulars
1	**Description the course :** **Including but Not limited to :**	India is growing rapidly as a global super-power. To face the challenges of the century and to keep up with the pace of the world, maintaining health is of prime importance. Giving thrust to healthy society, Physical Education, Sports, Health & fitness and Yoga are of great significance in today's world. The Government of India insists on Physical Fitness, Mental Health and Overall Development of Personality for every citizen. However, creating efficient and skilled human resource in the field of Physical Education, Sports and Yoga is identified as the need of the hour. Thus, the Governments of India and Government of Maharashtra have included Physical Education, Sports and Yoga as a key area under the NEP 2020.
2	**Vertical :**	Co-Curricular
3	**Type :**	Activity Based
4	**Credit**:	2 credits
5	**Hours Allotted :**	30 Hours
6	**Marks Allotted**:	50 Marks
7	**Course Objectives**: The student should be able to: • **To create awareness about the negative effects of a sedentary lifestyle** and highlight the importance of exercise in preventing lifestyle diseases. • **To encourage the development of time management skills** for incorporating regular fitness activities into daily life across all age groups and genders. • **To introduce advanced yoga practices and meditation techniques** for enhancing physical health and managing stress, anxiety, and related conditions.	
8	**Course Outcomes**: **CO1:** Identify the effects of a sedentary lifestyle and explain the role of regular exercise in preventing lifestyle-related diseases. **CO2:** Apply time management strategies to incorporate fitness into daily routines across different age groups and genders.	

	CO3: Demonstrate proficiency in advanced yoga techniques, including pranayama, meditation, and therapeutic asanas for mental and physical well-being.
9	**Module :** This course focuses on the impact of a sedentary lifestyle and the importance of exercise in preventing lifestyle diseases like diabetes, obesity, and hypertension. It emphasizes time management and integrating fitness into daily routines for all age groups and genders. The module also introduces advanced yoga practices including pranayama, mindfulness meditation, and complex asanas. Students will explore the therapeutic use of yoga for stress, anxiety, and back pain. Overall, the course promotes holistic health through physical activity and mental well-being.
10	**Reference Books:** 1. McCall, T. (2007). Yoga as Medicine: The Yogic Prescription for Health and Healing. New York: Bantam Books. 2. 5. Brown, D. & Gerbarg, P.L. (2012). The Healing Power of the Breath. Boston: Shambhala Publications.
11	**Internal Continuous Assessment: 40 %(20 Marks)** • Yoga Report/Assignment
12	**External Assessment: 60% : (30 Marks)** • Scanned copy of the Advanced Yoga participation certificate. • A short report on Yogasana (handwritten or typed). *Note: Students participating in sports competitions conducted by University at State or National Level, students who have represented Mumbai University or College at Intercollegiate / Inter Zonal / West Zone Inter University / All India Inter University/ International tournament are exempt from submission of report.*

Letter Grades and Grade Points:

Semester GPA/ Program CGPA Semester/ Program	% of Marks	Alpha-Sign / Letter Grade Result		Grade Points
9.00-10.00	90.0-100	O	(Outstanding)	10
8.00-<9.00	80.0-<90.0	A+	(Excellent)	9
7.00-<8.00	70.0-<80.0	A	(Very Good)	8
6.00-<7.00	60.0-<70.0	B+	(Good)	7
5.50-<6.00	55.0-<60.0	B	(Above Average)	6
5.00-<5.50	50.0-<55.0	C	(Average)	5
4.00-<5.00	40.0-<50.0	P	(Pass)	4
Below 4.00	Below 40	F	(Fail)	0
Ab (Absent)	-	AB	(Absent)	0

Signature of BOS Chairman
Dr. Bhakti Chaudhari
BOS (Computer Science)

Signature of Faculty Member
Ms. Rhucha Patil
(Computer Science)

Signature of I/C Principal
Dr. Swiddle D'Cunha