Pokhara University
Faculty of Science and Technology

Course Code: CMP 344 (3 Credits) | Full marks: 100
Course title: **Computer Networks (3-1-2)** | Pass marks: 45
Nature of the course: Theory & Practical | Total periods: 45
Level: Bachelor | Program: BE

1. **Course Description**

   This course is designed to familiarize the student with the basic taxonomy and terminology of the computer. It aims to provide an understanding about the operation of layer-wise network communication, various addressing mechanisms, routing algorithms, network management & security in the computer network and overview of server configuration for complete networking systems.

2. **General Objectives**

   The course is designed with the following objectives:

   - ➔ To acquaint the students with the computer networking concepts, including fundamental principles, terminology, and architecture.
   - ➔ To make the students familiar with the various network models and protocols at different layers, understanding their roles, functions, and how they enable communication between devices.
   - ➔ To expose the students to key concepts in network security, including strategies to protect data integrity, confidentiality, and availability, and to mitigate threats like hacking and data breaches.
   - ➔ To equip the students with the practical skills to design, configure, manage, and troubleshoot networks, including the use of networking tools, hardware, and software.

3. **Contents in Detail**

Specific Objectives	**Contents**

● The student will be able to understand the computer networking concepts, including fundamental principles, terminology, and architecture.	**Unit I: Introduction to Computer Network (3 hrs)** 1.1 Definition, merits, Demerits 1.2 Network Models 1.2.1 PAN, LAN, Campus Area Network (CAN), MAN, Country Area Network (CAN*), WAN, GAN 1.2.2 Topological Models (star, bus, distributed bus, mesh, tree, hybrid, ring) 1.2.3 Client/Server, Peer-to-Peer 1.3 ISPs, NSPs Overview and Backbone of Networking 1.4 Recent Trends in Telecom Technologies: 2G/3G/4G/5G.
● Understand the layered approach to networking and the various network connecting devices	**Unit II: Reference Model (4 hrs)** 2.1 Protocols and Standards 2.2 Interfaces and Services 2.3 OSI Layers 2.4 TCP/IP Layers 2.5 Comparison between OSI and TCP/IP 2.6 Networking hardware: NIC, Hub, Repeater, Switches, Bridge, Router, Gateway
● Understand how the Physical Layer establishes the foundation for all subsequent layers of the networking model, ensuring that data can be physically transmitted between devices effectively and reliably. ● Alongside, students will learn about the various networking parameters	**Unit III: Physical Layer (4 hrs)** 3.1 Guided Media: Copper, Fiber cabling and its capacity standards 3.2 Unguided Media: Bluetooth, Wi-Fi/Wireless LAN, Satellite Communication Basics (Microwaves, Radio waves) 3.3 Circuit/packet/message switching 3.4 ISDN signaling and Architecture 3.5 Network Performance: Bandwidth, Throughput, Latency, Bandwidth-Delay Product, Jitter

• In this chapter the student will learn how the data link layer provides reliable data transfer across a physical network link by handling error detection, frame synchronization, and flow control between directly connected devices.	**Unit IV: Data Link Layer (8 hrs)** 4.1 LLC and MAC sub-layer overview 4.2 Physical (MAC) addressing overview 4.3 Framing 4.4 Flow Control (stop and wait, go-back-N, selective-repeat-request) 4.5 Error Control Mechanism 4.5.1 Error Detection: Parity Check, CRC 4.5.2 Error Correction: Hamming Code 4.6 Channel Access 4.6.1 ALOHA Systems 4.6.2 CSMA, CSMA/CD 4.7 802.3 Ethernet, Fast Ethernet, Gigabit Ethernet 4.8 802.4 Token Bus, 802.5 Token Ring 4.9 Virtual Circuit Switching: Frame Relay, ATM and X.25
• Gain a good understanding of Internet Layer Protocol for ensuring that data packets are correctly routed and delivered across networks, using IP addresses.	**Unit V: Network Layer Protocols and Addressing (8hrs)** 5.1 Logical Addressing 5.1.1 IPV4 addressing, subnetting, supernetting, CIDR, VLSM 5.1.2 IPV6 addressing overview 5.1.3 IPV4 and IPV6 header protocol format 5.1.4 IPV4 and IPV6 feature comparison 5.2 Routing Algorithm Overview 5.2.1 Classful and Classless Routing 5.2.2 Adaptive and non-adaptive Routing 5.2.3 Distance vector and Link-state routing 5.2.4 Interior and exterior routing 5.2.5 Unicast and multicast routing 5.2.6 Routing Algorithms: RIP, OSPF, BGP 5.3 NAT
• Understand the concept of transport layer protocol to ensure reliable and efficient data transfer between devices by managing end-to-end communication.	**Unit VI: Transport Layer and Protocols (4 hrs)** 6.1 Port addressing overview 6.2 Process to process delivery: multiplexing and demultiplexing 6.3 TCP services, features, segment headers, well known ports & Handshaking 6.4 UDP services, features, segment headers, well known ports 6.5 Concept of socket programming: TCP and UDP socket

● In this chapter, the students will learn the traffic shaping algorithms used in computer networks to control the amount and rate of data transmission, helping to manage congestion and ensure QOS.	**Unit VII: Congestion Control and Quality of Services (3 hrs)** 7.1 Congestion Control: Open Loop and Closed Loop 7.2 Traffic Shaping (Leaky bucket and Token bucket) 7.3 TCP Congestion Control
● Learn how the Application Server Protocols facilitates communication between the application server and client devices, ensuring the efficient, secure, and reliable delivery of application services.	**Unit VIII: Application Layer, Servers and Protocols (4 hrs)** 8.1 Domain addressing, DNS server and Queries 8.2 HTTP, FTP & proxy server overview 8.3 DHCP Principles 8.4 Email Server Protocols: SMTP, POP, IMAP
● Here the student will learn how to protect the network and its data from unauthorized access, attacks, and breaches - ensuring confidentiality, integrity, and availability of information.	**Unit IX: Network Management and Security (7 hrs)** 9.1 Introduction to Network Management 9.2 Principles of Cryptography (Symmetric Key: DES, Asymmetric key: RSA) 9.3 Key Exchange Protocols (Diffie-Hellman, Kerberos) 9.4 VPN 9.5 Overview of IP Security 9.6 Firewall, Digital Certificate 9.7 Next Generation Network (NGN)

Note: The figures in the parentheses indicate the approximate periods for the respective units.

4. **Methods of Instruction**
   Lecture, Tutorials, Discussions and Assignments

5. List of Tutorials

The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover all the required contents of this course.

S.N.	Tutorials
1	Error Detection and Correction Methods, Parity ,CRC and Hamming code
2	Subnetting
3	Leaky Bucket and Token Bucket/ Queuing Delay Numericals
4	RSA and cryptography Numerical

6. Practical Works

S.N.	Practical works
1	Network commands testing: ping-pong, netstat, nslookup, ipconfig/ifconfig, tracert/traceroute.
2	Setting up Client/Server network system in Microsoft and Linux environment
3	UTP CAT6 cabling: Straight and Cross wiring, testing and verification
4	Internet Packet header analysis using TCPDUMP/WIRESHARK
5	Router Configuration use of packet tracer or other simulator software
6	OSPF configuration and practices
7	VLAN And Router on stick method
8	Web, Proxy, FTP server configuration
9	Implementation of Router ACL, Proxy Firewall, IPTables
10	Case Study: Network Design Standards (eg: building network design with servers including NCR

7. Evaluation system and Students' Responsibilities

Evaluation System

In addition to the formal exam(s) conducted by the Office of the Controller of Examination of Pokhara University, the internal evaluation of a student may consist of class attendance, class participation, quizzes, assignments, presentations, written exams, etc. The tabular presentation of the evaluation system is as follows.

Internal Evaluation	Weight	Marks	External Evaluation	Marks
Theory		**30**	**Semester End Examination**	**50**
Class attendance and participation	10%			
Assignments	20%			
Quizzes/ presentations	10%			
Internal Term Exam	60%			
Practical		**20**		
Attendance and class participation	10%			
Lab Report/Project Work	20%			
Practical Exam/Project Work	40%			
Viva	30%			
Total Internal		**50**		
Full Marks = 50 +50 =100				

Students' Responsibilities:
Each student must secure at least 45% marks in the internal evaluation with 80% attendance in the class to appear in the Semester End Examination. Failing to obtain such a score will be given NOT QUALIFIED (NQ) and the student will not be eligible to appear in the End-Term examinations. Students are advised to attend all the classes and complete all the assignments within the specified time period. Students are required to complete all the requirements defined for the completion of the course

8. **Prescribed Books and References**

Text Book

1. "Computer Networks", 4th Edition, A. S. Tanenbaum, Pearson Education.
2. "Data Communications and Networking", 5th Edition, Behrouz A. Forouzen, McGraw-Hills.

Reference Books

1. "Data & Computer Communications", 7th Edition, William Stallings, Pearson Education.
2. "Computer Networking: A Top-Down Approach", James F. Kurose, K.W. Rose, 6th Edition, Pearson Education.

Pokhara University
Faculty of Science and Technology

Course No.: CMP 338
Course title: **Simulation and Modeling**
Nature of the course: Theory and Practical

Level: Undergraduate

Full marks: 100
Pass marks: 45
Time per period: 1 hour
Total periods: 45
Program: BE

1. Course Description

This course covers the various concepts system simulation. This course emphasizes on fundamental concept, principles and properties of continuous system and discrete system. It covers examples, solutions and programming language regarding continuous and discrete system. It also covers probability concepts and random number generation technique and testing. Output generated from the process is analyzed.

2. General Objective

The main objectives of the course are

- To provide basic knowledge of various systems.
- To study continuous and discrete system.
- To get the concept of probability concept and random numbers.

3. Methods of Instruction

3.1.General instructional Techniques: Lecture, discussion, readings.
3.2.Specific instructional Techniques: Lab works, Project works

4. Contents in Detail

Specific Objectives	Contents
• Familiarize and compare the various concept of system and its environment. • To explain why simulations are used in systems analysis and design, emphasizing their role in modeling complex systems.	**Unit 1 : Introduction to simulation and modeling (4 hrs)** 1. System and its concept 2. System Environment 3. Types of System (continuous and discrete, static and dynamic, stochastic and deterministic) 4. Steps of Simulation 5. Advantage, disadvantage and application of simulation 6. System Modeling and types of models 7. Principles of Modeling 8. Verification and validation of model

• To introduce simulation techniques such Monte Carlo simulation. • To provide knowledge on how to create accurate models that represent real-world systems.	**Unit 2 : System Simulation(8 hrs)** 2.1Monte Carlo Method 2.1.1 Problems regarding Monte Carlo method 2.2 Comparison of simulation and analytic solution 2.3 System simulation and its types 2.4 Real time simulation 2.5 Lag Models (Distributed lag Model, Cobweb Model) 2.6 Queuing system and its characteristics and notation 2.7 Single server queuing model 2.7.1 Arrival routine 2.7.2 Departure routine 2.7.3 Performance measure of SSQM 2.8 Time advance mechanism (next event oriented and fixed increment oriented)
• To represent continuous system using differential equations and other mathematical tools. • To analyze system dynamics, understand their stability, and predict future behavior. • To solve and implement continuous system using analog method and programming language.	**Unit 3 : Continuous System(8 hrs)** 3.1 Introduction to continuous system 3.2 Representation of continuous system using differential equation 3.3 Linear and nonlinear differential equations and its examples 3.4 Analog Computer (Components and examples) 3.5 Digital Analog Simulators 3.6 Hybrid Computers 3.7 CSSLs, CSMP III 3.7.1 Structural Statement 3.7.2 Data Statements 3.7.3 Control Statements 3.8 Feedback System with example 3.9 Interactive System
• For understanding and modeling processes where changes occur at distinct, separate points in time or involve discrete states. • To gather statistics white studying discrete system.	**Unit 4 : Discrete System(7 hrs)** 4.1 Introduction to discrete system 4.2 Components of discrete system 4.3 Representation of Time 4.4 Examples for discrete system 4.4.1Telephone call system as lost call and delayed call system 4.4.2 Bank Queue System 4.5 Simulation Programming Task 4.6 Steps of simulation programming task 4.7 Gathering Statistics 4.7.1 Counters and Summary Measures 4.7.2 Measuring Utilization and Occupancy 4.7.2 Recording Distribution and Summary Measures 4.8 Discrete System Simulation Languages

● To understand probability distributions and random variables, which can accurately represent random phenomena in simulations. ● To generate random numbers using various generators and test their independence and uniformity property.	**Unit 5 : Probability Concept and Random Numbers(7 hrs)** 5.1 Stochastic System 5.2 Discrete and continuous probability function 5.3 Random numbers versus pseudo random numbers 5.4 Properties of random numbers 5.5 Random number generation Techniques 5.5.1 Linear Congruential Generator 5.5.2 Mixed Generator 5.5.3 Additive and Incremental Generator 5.6 Test for randomness 5.6.1 Uniformity Test - KS Test - Chi Square Test 5.6.2 Independence Test - Run Test (above and below, up and down, length of Runs - Test for Auto correlation - Gap Test - Poker Test
● To test different scenarios, identifying potential issues, and optimizing performance without the risks and costs associated with real-world trials.	**Unit 6 : Discrete System Languages (6 hrs)** 6.1 Simulation using GPSS 6.1.1 GPSS problems 6.2 Simulation using SIMSCRIPT 6.2.1 Organization of SIMSCRIPT 6.2.2 Programs of SIMSCRIPT 6.3 Other discrete simulation Languages
● To interpret, understand, and make decisions based on the results generated by a simulation model. ● To validate the simulation results, optimizing system performance, and providing actionable insights.	**Unit 7 : Output Analysis Method(5 hrs)** 7.1 Nature of Problem 7.2 Estimation Method 7.3 Simulation Run Statistics 7.4 Replication of Runs 7.5 Elimination of Initial Bias

5. Laboratory work: (30 hrs)

1. Representing ohm's law and verifying its VI characteristics.
2. Generating value of pi using Monte Carlo method and check its accuracy level
3. Implementing various models in simulation
4. Generating random numbers and their testing

5. Implementing GPSS programs
6. Examples of continuous and discrete system
7. Develop a small project to simulate any mathematical model

6. **List of Tutorials:**
   The various tutorial activities that suit this course should cover all the content of this course to give student a space to engage more actively with the course content in the presence of instructor. Students should submit tutorials as assignments or class works to the instructor for evaluation. The following tutorial activities of 15 hours should be conducted to cover all the content of course:

   A. Discussion-based Tutorials: (6 hrs)
      1. Explain the concepts of system modeling, abstraction, and the simulation life cycle.
      2. Example of different models that can be simulated.
      3. Analyzing the output obtained from simulation.
      4. Continuous and discrete system example.
   B. Problem solving-based Tutorials: (9 hrs)

      1. Examples using Monte Carlo simulation technique.

      2. Example questions for distributed lag model and cobweb model.

      3. Numerical to generate random numbers.

      4. Testing random number properties using various techniques.

7. **Evaluation system and Students' Responsibilities**

Internal Evaluation

In addition to the formal exam(s), the internal evaluation of a student may consist of quizzes, assignments, lab reports, projects, class participation, etc. The tabular presentation of the internal evaluation is as follows.

External Evaluation	**Marks**	**Internal Evaluation**	**Weight**	**Marks**
Semester-End examination	50	Assignments	12%	
		Attendance	6%	
		Unit test	14%	
		Assessment	28%	
		Practical	40%	
Total External	50	Total Internal	100%	50
Full Marks 50+50 = 100				

Student Responsibilities:

Each student must secure at least 45% marks in internal evaluation with 80% attendance in the class in order to appear in the Semester-End Examination. Failing to get such score will be given NOT QUALIFIED (NQ) and the student will not be eligible to appear the Semester-End Examination. Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during the period. If a student fails to attend a formal exam, test, etc. there won't be any provision for re-exam.

8. Prescribed Books and References

Text Book

1. G Gorden, **System Simulation**, Prentice Hall of India.
2. Jerry Banks, John S. Carson II, Barry L Nelson, David M. Nicol, **Discrete Event System Simulation,**

Reference Books

1. "Simulation Modeling and Analysis" by Averill M. Law and W. David Kelton
2. "System Simulation with Digital Computer" by N. W. McCormick
3. "Simulation and the Monte Carlo Method" by Reuven Y. Rubinstein and Dirk P. Kroese

Pokhara University
Faculty of Science and Technology

Course Code: CMP 226 (3 Credits) Full Marks: 100

Course title: Applied Operating Systems (3-1-2) Pass Marks: 45

Nature of the course: Theory and Practical Total Lectures: 45

hrs Level: Bachelor Program: BE (Software/IT)

1. Course Description

This course is designed to visualize and apply the concepts of operating systems to various system environments. It includes the system software, internal structure, choice and security features of the operating system for running the application software and performing various tasks with optimized throughput. This course also introduces the emerging new trending operating system for distributed environments like cloud systems. This course aims to provide the students with the basic concepts of design and development of an operating system applying necessary security and safety measures.

2. General Objective

- To acquaint the students with the knowledge of the structure of operating systems and their functionality.
- To develop the skills in students to select and design optimal resource allocation schedules.
- To acquaint the students with the knowledge of process and thread, I/Os, Memory, CPU, disk management.
- To acquaint the students with basic concepts of security and safety measures in operating systems.

3. Methods of Instruction

- Lectures, Tutorials, Case Studies, Discussion, Readings and Practical Works.

4. Contents in Detail

Specific Objectives	Contents
• Familiarize with basic concepts of Operating system, and its architecture. • Understanding the successive evolution and structure of operating system	**Unit 1. : Introduction [5 hours]** 1.1 Applied Operating System 1.2 Goals of an Operating System 1.3 Computer System Organization and Architecture 1.4 Operating-System Structure 1.5 Generations of Operating Systems (0th, First, Second, Third, Fourth) 1.6 Types of Operating Systems 1.6.1 Batch Processing Operating System 1.6.2 Multiprogramming Operating System

	1.6.3 Multiprocessing System 1.6.4 Time Sharing 1.6.5 Real Time Operating System (RTOS) 1.6.6 Networking Operating System 1.6.7 Distributed Operating System 1.6.8 Operating Systems for Embedded Devices 1.7 Functions of OS 1.8 Kernel Data Structures 1.9 User and Operating-System Interface 1.10 System Call, Types of System Calls and Handling System Call, 1.11 Virtual Machine
• Familiarize with the concept of Process and threads • Understanding the mutual exclusion for resource utilization • Mastering Process Scheduling Algorithms. • Dealing with Deadlocks and Resource Management:	**UNIT- II: Process, Threads and Scheduling** [13 hours] 2.1 Process 2.1.1 Process Concept 2.1.2 Process State 2.1.3 Operations on Processes 2.1.4 Process Control Block 2.1.5 Implementation of Processes 2.2 Threads 2.2.1 Thread Concept and Usage 2.2.2 Threads Models (Many-to-one model, One-to-One Model, Many-to many model) 2.2.3 User Space and Kernel Space Threads 2.2.4 Hybrid Implementations 2.2.5 Threads Vs Processes 2.3 Inter-process Communication and Synchronization 2.3.1 Cooperating Processes 2.3.1 Race Conditions 2.3.2 Critical-Section Problem 2.3.3 Mutual Exclusion with Busy Waiting (Disabling Interrupt, Lock variable, Strict alternation, Peterson's Algorithm, Test and Set Lock) 2.3.4 Sleep and Wakeup 2.3.5 Semaphores 2.3.6 Mutexes 2.3.7 Monitors 2.3.8 Message Passing 2.3.9 Classical IPC Problems 2.4 Process Scheduling 2.4.1 Basic Concept and Types(Preemptive, Non-Preemptive). 2.4.2 Scheduling Criteria 2.4.3 Scheduling Algorithm (First come first served Scheduling(FCFS), Round-Robin,, Shortest Job First(SJF), Shortest process next, Shortest Remaining Time First(SRTF), Priority Scheduling, Priority Fair Share, Real Time Scheduling, Guaranteed, Lottery Scheduling)

	2.5 Thread Scheduling 2.6 Deadlocks 2.5.1System Resources: Preemptable and Non-preempable 2.5.2Methods of Handling Deadlock(DeadlockPrevention, Deadlock Avoidance: Banker's Algorithm, DeadlockDetection: Resource allocation graph, Recovery from Deadlock) 2.5.3Security and Protection- Security Threats, System Protection, Goals of Protection, Principles of Protection, Domain of Protection, Access Matrix, Implementation of Access Matrix, access Control list
• Conceptualize the role and working procedure of memory • Familiarizing with virtual memory management • Understanding the page replacement algorithms • Understand the mechanism file and filing	**UNIT- III: Storage Management** [12 hours] 3.1 Memory Management 3.1.1 Logical & physical Address Space 3.1.2 Swapping 3.1.3 Contiguous Memory Allocation 3.1.4 Paging, Structure of Page Table 3.1.5 Segmentation, Segmentation with Paging 3.2 Virtual Memory 3.2.1 Background 3.2.2 Demand Paging, 3.2.3 Performance of Demanding Paging, 3.2.4 Page Replacement, Page Replacement Algorithms, Allocation of Frames, 3.2.5 Thrashing. 3.3. File System Interface and Implementation 3.3.1 File System Interface (The Concept of a File, Access methods, Directory Structure, File System Mounting, File Sharing, Protection) 3.3.2 File System Implementation (File System Structure, File System Implementation, and Allocation methods, Free-space Management, Directory Implementation, Efficiency and Performance)
• Understand the role of input/output devices • Understand the different approaches for optimal output	**Unit IV : Input/output Management** [7 hours] 4.1 I/O System 4.1.1 Principles of I/O Hardware (I/O Device, Device Controller, Memory Mapped I/O, Direct Memory Access) 4.1.2 Principles of I/O Software (Goals of I/O Software, Polled I/O vs Interrupt Driven I/O, Character User Interface and Graphical User Interface, Device Driver, Device Independent I/O Software, User-space I/O Software 4.2 Mass Storage Structure - Overview of Mass Storage Structure, Disk Structure, Disk Attachment, Disk Scheduling, Disk Management, Swap space Management Redundant Array of Inexpensive Disks
• Visualize of operating system structure and functionality	**Unit V: Case Study** [5 hours] 5.1 Linux - Design principles, Inter-process communication, Kernel modules, Network structure, Security 5.2 Windows **-** Design principles, Programmer interface, System

	components, Security level
• Familiarize with current trends of operating systems	**Unit VI : Emerging Trends in Operating System** [3 hours] 6.1 Concept, Character and Role of Distributed Operating System and Cloud Operating System 6.2 Security issues and method of deployment 6.3 Memory wall and bottleneck for operating system

5. Practical Works

Laboratory work of 30 hours per group of maximum 24 students should cover the operating system structure and functions of any two popular operating systems. It also insists the students to design a model of operating system with the reference of open source guidelines. Students should complete the following tasks in laboratory:

1. Understanding and running the internal commands and external commands in MS DOS.
2. Installation and user, application management in Windows (current version)
3. Simulation of Process Scheduling Algorithms
4. Simulation of Page Replacement Algorithms
4. Simulation of Disk Arm Scheduling Algorithms
5. System Administration (user, disk, role, etc.) in any open source operating system.

6. List of Tutorials

The various tutorial activities that suit this course should cover all the content of this course to give students a space to engage more actively with the course content in the presence of the instructor. Students should submit tutorials as assignments or class-works to the instructor for evaluation. The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover the content of this course:

A. Discussion-based Tutorials: (2 hrs)

1. Operating System Services (Class discussion)
2. Deadlock and Starvation
3. New Trends in Operating Systems (Oral Presentation).

B. Problem solving-based Tutorials: (10 hrs)

1. Evaluating the performance of different Process Scheduling Algorithm, Disk Scheduling Algorithm and Page Replacement Algorithm through hypothetical scenarios.
2. WAP that creates multiple processes and manages their execution using different scheduling algorithms such as Round Robin, Priority Scheduling, and Shortest Job First.
3. Implementing page replacement algorithms such as FIFO, LRU, and Clock, and evaluating their performance in a simulated virtual memory environment.
4. WAP that simulates FCFS, SCAN and CSCAN disk scheduling algorithms.
5. Develop basic shell scripts with defined functions, perform simple tests, use loops for iteration, create patterns, utilize expansions, and handle substitutions within shell programming.

6. Develop a program in UNIX utilizing system calls such as fork, exec, getpid, exit, wait, close, stat, opendir, and readdir to perform various operations within the operating system.

C. Review and Question/Answer-based Tutorials: (3 hrs)

1. Case study of "Linux and Windows Operating System" followed by Oral Presentation in class.
2. Students ask questions within the course content and assignments and review key course content in preparation for tests or exams.

7. **Evaluation System and Students' Responsibilities**

Internal Evaluation

The internal evaluation of a student may consist of assignments, attendance, internal assessment, lab reports, project works etc. The internal evaluation scheme for this course is as follows:

Internal Evaluation	Weight	Marks	External Evaluation	Marks
Theory		30	Semester-End examination	50
Attendance & Class Participation	10%			
Assignments	20%			
Presentations/Quizzes	10%			
Internal Assessment	60%			
Practical		20		
Attendance & Class Participation	10%			
Lab Report/Project Report	20%			
Practical Exam/Project Work	40%			
Viva	30%			
Total Internal		50		
Full Marks: 50 + 50 = 100				

Student Responsibilities

Each student must secure at least 45% marks separately in internal assessment and practical evaluation with 80% attendance in the class in order to appear in the Semester End Examination. Failing to get such a score will be given NOT QUALIFIED (NQ) to appear for the Semester-End Examinations. Students are advised to attend all the classes, formal exam, test, etc. and complete all the assignments within the specified time period. Students are required to complete all the requirements defined for the completion of the course.

9. Prescribed Books and References

Text Books

1) A. Silberschatz, P.B. Galvin, G. Gagne. (2009). *Applied Operating System Concepts*. John Wiley & Sons, Inc.

References

1) Tanenbaum A.S. (2011). *"Modern Operating System"*, Pearson.
2) Dhamdhere D. M. (1999). *System Programming and Operating System*. Tata McGraw-Hill.
3) Tanenbaum A. S. (2017). *Distributed Operating System*. Pearson.

Pokhara University
Faculty of Science and Technology

Course No.: CMP 342 (3 Credits)	Full marks: 100
Course title: Artificial Intelligence and Neural Network (3-1-3)	Pass marks: 45
Nature of the course: Theory and Practical	Total Lectures: 45 hrs
Level: Bachelor	Program: BE (Software)

1. Course Description

This course is designed to provide an in-depth study of Artificial Intelligence (AI) with a special focus on neural networks and their applications in software engineering. The course covers foundational AI concepts, problem-solving strategies, knowledge representation and expert systems. Students will also explore the design, implementation, and application of neural networks to solve complex software engineering problems.

2. General Objectives

- To provide the students with the foundational principles and techniques of AI and neural networks.
- To develop the skills in students to design and implement AI-based solutions in software engineering.
- To acquaint the students with the knowledge of various AI domains such as knowledge representation, expert systems and neural network.
- To acquaint the students with the knowledge of advanced topics in neural networks, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).
- To provide the students with the knowledge to critically evaluate the ethical implications of AI and its impact on society.

3. Methods of Instruction

Lecture, Discussion, Readings, Practical works and Project works.

4. Contents in Detail

Specific Objectives	Contents

• Explain artificial intelligence, its approaches and its foundations. • Critically evaluate the ethical implications of AI and its impact on society.	**1. Introduction to Artificial Intelligence (4 hrs)** 1.1. Intelligence 1.1.1. Types of Intelligence 1.1.2. Components of Intelligence 1.2. Artificial Intelligence 1.2.1. Approaches of AI 1.2.1.1. Acting Humanly 1.2.1.2. Thinking Humanly 1.2.1.3. Thinking Rationally 1.2.1.4. Acting Rationally 1.2.2. Foundations of AI 1.2.3. History of AI 1.2.4. Risk and Benefits of AI 1.3. Ethics and Societal Implications 1.3.1. Ethical Implications of AI 1.3.2. AI and Society: Work and Automation, Employment, Privacy and Security 1.3.3. Governance and Regulation
● Design and implement intelligent agents.	**2. Intelligent Agents (5 hrs)** 2.1. Agents and Environments 2.2. Concept of Rationality 2.2.1. Performance Measures 2.2.2. Rationality and Rational Agent 2.3. Task environment and its properties 2.4. Structure of Agents 2.4.1. Agent programs 2.4.2. Types of agent programs 2.5. Learning Agents
● Formulate the real world problems and apply the search algorithms to solve them.	**3. Problem Solving and Search Algorithms (8 hrs)** 3.1. Problem Solving 3.1.1. Problem Solving Agents 3.1.2. Problem solving process 3.1.3. Production System 3.1.4. Well-defined and ill-defined problems 3.1.5. Problem formulation 3.2. Search Algorithms 3.2.1. Uninformed Search 3.2.1.1. Breadth- First Search 3.2.1.2. Depth-First Search 3.2.1.3. Iterative Deepening Search 3.2.2. Informed Search 3.2.2.1. Heuristics 3.2.2.2. Greedy Best-First Search 3.2.2.3. A* Search 3.3. Local Search and Optimization Problems 3.3.1. Hill-Climbing Search and its problems (Local maxima, plateaus, and ridges) 3.3.2. Simulated Annealing

	3.3.3. Genetic Algorithms 3.4. Adversarial Search and Game Playing 3.4.1. Minimax algorithm 3.4.2. Alpha-beta pruning
● Represent the knowledge of a domain and apply inference rules to draw conclusions.	**4. Knowledge Representation and Reasoning (4 hrs)** 4.1. Overview of Propositional and PredicateLogic 4.1.1. Syntax 4.1.2. Semantics 4.1.3. Inference and Resolution 4.2. Reasoning Under Uncertainty 4.2.1. Probabilistic Reasoning 4.2.1.1. Bayesian Networks 4.3. Other Approaches to Knowledge Representation 4.3.1. Semantic Nets and Frames 4.3.2. Ontological-Based Representation
● Design and develop an expert system to solve a real-world problem.	**5. Expert System (4 hrs)** 5.1. Definition and History of Expert System 5.2. Architecture of Expert Systems 5.3. Knowledge Representation in expert system 5.3.1. Logic based representation 5.3.2. Rule-based system 5.3.3. Semantic networks 5.3.4. Ontology-based Systems 5.3.5. Frame-based Systems 5.4. Inference Mechanisms 5.4.1. Forward Chaining 5.4.2. Backward Chaining 5.5. Knowledge Acquisition and Learning 5.6. Applications of Expert Systems
● Design, implement and evaluate neural networks. ● Apply the neural Network for software testing and debugging.	**6. Artificial Neural Network (12 hrs)** 6.1. Introduction to Machine Learning 6.2. Types of Machine Learning 6.2.1. Supervised Learning 6.2.2. Unsupervised Learning 6.2.3. Reinforcement Learning 6.3. Introduction to Neural Network 6.3.1. Biological inspiration: neurons and synapse 6.3.2. Structure of a Neuron 6.3.2.1. Artificial Neural Network 6.3.2.2. Components: weights, biases and activation functions 6.3.3. Neural Network Architectures 6.3.3.1. Feedforward 6.3.3.2. Convolution 6.3.3.3. Recurrent 6.3.4. Perceptrons 6.3.4.1. Single layer perceptron

	6.3.4.2. Multilayer Perceptron 6.3.4.3. Backpropagation 6.4. Training Neural Network 6.4.1. Forward and Backward Propagation 6.4.1.1. Forward Propagation 6.4.1.2. Backpropagation and Gradient Descent 6.4.2. Loss functions 6.4.2.1. Role of loss function 6.4.2.2. Mean Squared Error (MSE) 6.4.2.3. Cross-Entropy Loss 6.4.3. Regularization Techniques 6.4.3.1. Overfitting and underfitting 6.4.3.2. Regularization methods: L1, L2, Dropout, Batch Normalization 6.5. Neural Network Optimization Techniques 6.5.1. Gradient Descent 6.5.2. Adam OPtimizer 6.6. Model Evaluation 6.6.1. Cross-validation, confusion matrix, 6.6.2. Precision, recall, F1 -score 6.7. Neural Networks in Software Engineering 6.7.1. Neural Networks for Software Testing and Debugging 6.7.2. AI-Driven Code Generation and Optimization 6.7.3. Neural Networks in Automated Software Maintenance
● Design and implement Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).	**7. Deep Learning (8 hrs)** 7.1. Convolution Neural Network (CNNs) 7.1.1. CNNs and their components 7.1.2. Convolution, Pooling and fully connected layers 7.1.3. Applications in image processing and computer vision 7.2. Recurrent Neural Networks (RNNs) 7.2.1. Basics of RNNs 7.2.2. Long Short-Term Memory (LSTM) 7.2.3. Gradient Recurrent Units (GRU) 7.2.4. Applications in time-series prediction 7.3. Advanced Topics in Neural Network 7.3.1. Transfer Learning and Pretrained Models 7.3.2. Generative Adversarial Networks (GANs) 7.3.3. Reinforcement Learning and Neural Network

5. Practical Works

Laboratory work of 45 hours per group of maximum 24 students should cover implementation of the following lab works:

SN	Implementation Description
1	Implement search algorithms (e.g., BFS, DFS, A*) in Python.
2	Develop a simple expert system using rule-based reasoning.
3	Build and train CNNs for image classification.
4	Build and train RNNs for sequence prediction.
5	Design a neural network-based tool for a specific software engineering task (e.g., bug detection, code optimization).

Students should submit a project work that uses all the knowledge obtained from this course to solve any problem chosen by themselves. The marks for the practical evaluation must be based on the project work submitted by students.

6. List of Tutorials

The various tutorial activities that suit your course should cover all the content of the course to give students a space to engage more actively with the course content in the presence of the instructor. Students should submit tutorials as assignments or class works to the instructor for evaluation. The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover the content of this course:

A. Discussion-based Tutorials: (3 hrs)
   a. Evolution of Artificial Intelligence (Class discussion).
   b. AI and Society: Employment, Privacy and Security (Class discussion).

B. Problem solving-based Tutorials: (6 hrs)
   a. Apply a search technique to solve a real world problem.
   b. Design and develop a simple expert system that solves a real world problem.

C. Review and Question/Answer-based Tutorials: (6 hrs)
   a. A detailed case study on Tools and Frameworks for implementing neural networks: TensorFlow, PyTorch and Python (Oral Presentation in class).
   b. Case study on real-world applications of Neural Network in Software Engineering
   c. Students ask questions within the course content, assignments and review key course content in preparation for tests or exams.

7. Evaluation System and Students' Responsibilities

Evaluation System

The internal evaluation of a student may consist of assignments, attendance, internal assessment, lab reports, project works etc. The internal evaluation scheme for this course is as follows:

Internal Evaluation	Weight	Marks	External Evaluation	Marks
Theory		30	Semester-End examination	50
Attendance & Class Participation	10%			
Assignments	20%			
Presentations/Quizzes	10%			
Internal Assessment	60%			
Practical		20		
Attendance & Class Participation	10%			
Lab Report/Project Report	20%			
Practical Exam/Project Work	40%			
Viva	30%			
Total Internal		50		
Full Marks: 50 + 50 = 100				

Student Responsibilities

Each student must secure at least 45% marks separately in internal assessment and practical evaluation with 80% attendance in the class in order to appear in the Semester End Examination. Failing to get such a score will be given NOT QUALIFIED (NQ) to appear for the Semester-End Examinations. Students are advised to attend all the classes, formal exam, test, etc. and complete all the assignments within the specified time period. Students are required to complete all the requirements defined for the completion of the course.

8. Prescribed Books and References

Text Books

1. Russell, S. J., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
2. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

References

1. Ian, G. (2016). *Deep learning/Ian Goodfellow, Yoshua Bengio and Aaron Courville*. The MIT Press.
2. Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow.* O'Reilly Media, Inc.
3. Aggarwal, C. C. (2018). *Neural Networks and Deep Learning*. Springer.

Pokhara University
Faculty of Science and Technology

Course Code.: CMP 336
Course title: **Data Science and Machine Learning (3-1-2)**
Nature of the course: Theory & Practical
Year, Semester:……
Level: Bachelor

Full marks: 100
Pass marks: 45
Time per period: 1 hour
Total periods: 45
Program: BE Software

1. **Course Description**

This course provides a comprehensive introduction to the fields of Data Science and Machine Learning, aimed at equipping students with the essential knowledge and practical skills required to analyze data, interpret data, apply machine learning methods and visualize results.

It will include the following information:

- Covers a wide range of topics, including data pre-processing, statistical analysis, machine learning algorithms, model evaluation, and the application of these techniques to solve real-world problems.
- Delivery approach includes hands-on labs, case studies and deep understanding of how to leverage data to make informed decisions.

2. **General Objectives**

The course is designed with the following general objectives:

- To provide students with a foundational understanding of Data Science and Machine Learning.
- To familiarize students with techniques for cleaning, transforming, and visualizing data to uncover patterns and insights.
- To provide the knowledge for use of mathematics such as statistics, probability for data analysis and machine learning,supervised learning algorithms, linear regression, decision trees, and support vector machines, and their applications.

- To expose students to unsupervised learning techniques such as clustering and dimensionality reduction, and their use in identifying patterns and simplifying data.

.

3. Contents in Detail

This section contains the details to be taught under the course.

Specific Objectives	Contents
● Intends to provide a brief introduction to the field of Data Science and Machine Learning. ● Learn about various domains within Data Science and how they interrelate. ● Helps students understand the significance of data in modern decision-making.	**Unit I: Introduction to Data Science and Machine Learning (4Hrs)** 1.1 Definition and Overview of Data Science and Machine Learning 1.2 Applications of Data Science in various industries 1.3 Types of Data: Clean Data and Dirty Data 1.3 Data Science, AI, and Machine Learning
● Intends to get students well-acquainted with data collection methods and preprocessing techniques. ● Able to apply various preprocessing techniques to clean and prepare data for analysis.	**Unit II: Data Collection and Preprocessing (7 Hrs)** 2.1 Different Data Collection Methods for Machine Learning: Surveys, Sensors, Web Scraping, APIs, Databases 2.2 Data Quality Issues: Missing Data, Noisy Data, Inconsistent Data, Data Transformation 2.3 Techniques for Handling Missing Data 2.4 Data Cleaning Techniques: Handling Outliers, Dealing with Categorical Data, Normalization, and Standardization 2.5 Dependent and independent variables
● Intends to provide students with the skills to explore and understand data. ● Learn about various EDA techniques to identify patterns and insights in the data	**Unit III: Exploratory Data Analysis (6Hrs)** 3.1 Introduction to EDA 3.2 Descriptive Statistics: Mean, Median, Mode, Standard Deviation, Variance, Skewness, Kurtosis 3.3 Data Visualization Techniques: Histograms, Box Plots, Scatter Plots, Heatmaps 3.4 Identifying Trends: Mann–Kendall, Spearman's Rank, Sen's Slope 3.5 Correlations 3.6 Introduction to Hypothesis Testing
● Intends to provide students with the skills to explore and understand data. ● Learn about various EDA techniques to identify patterns and insights in the data	**Unit IV: Data Engineering (5 Hrs)** 4.1 Data pipeline, Design and Monitoring 4.2 Extract, Transform and Load (ETL) 4.3 Feature Engineering 4. 5 Feature Selection 4.6 Dimensionality Reduction: PCA, LDA
● Helps students learn how to implement basic machine learning models. ● Able to differentiate between various machine learning	**Unit V: Introduction to Machine Learning (9 Hrs)** 5.1 Definition and Types of Machine Learning: Supervised, Unsupervised Learning, Reinforcement Learning 5.2 Overview of the Machine Learning

algorithms and their applications.	5.3 Supervised Learning: Linear Regression, Logistic Regression, Decision Trees, Random Forest, k-NN, Support Vector Machines (SVM) 5.4 Unsupervised Learning: k-Means Clustering, Hierarchical Clustering methods Key Concepts: Training, Testing, Validation, Overfitting, Underfitting
● Apply basic and machine learning methods for detecting anomalies.	**Unit VI: Anomaly Detection (4 Hrs)** 6.1 Definition 6.2 Types: point, contextual, collective 6.3 Applications 6.4 Techniques for Anomaly Detection 6.4.1 Statistical Methods 6.4.2 Distance-based Methods 6.4.3 Density-based Methods 6.4.4 Clustering based Methods 6.4.5 Common Methods (one-class classification, isolation forest) 6.5 Anomaly Detection in High-Dimension
● Intends to get students well-acquainted with model evaluation techniques. ● Able to make use of various optimization techniques to improve model performance.	**Unit VII: Model Evaluation and Optimization (6 Hrs)** 7.1 Confusion Matrix, 7.2 Evaluation Metrics 7.2.1 Supervised: Accuracy, Precision, Recall, F1 Score, ROC Curve, AUC, MSE, True Positive Rate, False Positive, MSE, MAE, RMSE 7.2.2 Unsupervised: Purity, Rand Index, Silhouette Coefficient, Dunn Index 7.3 Cross-Validation Techniques 7.4 Hyperparameter Tuning: Grid Search, Random Search 7.5 Model Selection Techniques: Bias-Variance Trade-off, Ensemble Methods (Bagging and Boosting) 7.6 SMOTE Technique to Handle Imbalance 7.7 Time & Space Complexity of Machine Learning Models
● Helps students understand the ethical implications and legal considerations in data science.	**Unit VIII: Ethical and Legal Considerations in Data Science (4 Hrs)** 8.1 Data Privacy and Security 8.2 Ethical Issues in Data Science: Bias, Transparency, Accountability 8.3 Legal Considerations: Data Protection Laws, Intellectual Property

4. Methods of Instruction

The course will utilize a mix of lectures, tutorials, case studies, and lab sessions to support learning. Lectures will deliver core knowledge, while tutorials and case studies will enhance

comprehension. Lab sessions will provide hands-on experience, enabling students to apply theory to practical, real-world situations. This integrated approach ensures a well-rounded learning experience, fostering both theoretical insight and practical skills essential for success in data science and analytics.

5. Case Studies

Students will complete the following case studies and submit their reports:

- Exploratory Data Analysis (Agricultural Commodities): Students will conduct a comprehensive exploratory data analysis on a dataset related to agricultural commodities. This will involve analyzing trends, patterns, and correlations to provide insights.
- Supervised Learning (Customer Churn Prediction in Telecommunications): Students will build and evaluate a supervised learning model to predict customer churn in the telecommunications industry. The case study will require them to preprocess data, select relevant features, and apply classification algorithms to identify customers at risk of leaving.
- Anomaly Detection in Real-World Applications: Students will implement anomaly detection techniques to identify unusual patterns or outliers in a real-world dataset. This case study will involve applying various anomaly detection methods to solve practical problems such as fraud detection or system monitoring.

Students are required to submit a detailed report documenting their approach, results, and analysis.

6. List of Tutorials

The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover all the required contents of this course.

S.N.	Tutorials
1	● Using libraries of your programming choices (e.g. pandas, R) to manipulate datasets. ● Conducting exploratory data analysis (EDA) on real-world datasets. ● Cleaning and preprocessing data to prepare for modeling.
2	● Solving problems related to descriptive statistics (mean, median, mode, variance). ● Applying probability concepts to data science problems. ● Working with probability distributions and sampling techniques.
3	● Solving problems involving matrix operations and vector calculus.
4	● Applying linear algebra concepts to data transformations. ● Implementing supervised models like linear regression, decision trees, and k-nearest neighbors.

	• Implementing unsupervised model like k-means, hierarchical • Implementing anomaly detection for real world data. • Understanding the concept of overfitting and underfitting through practical examples. • Hyperparameter tuning and model evaluation techniques.
6	• Creating visualizations using Matplotlib and Seaborn. • Visualizing complex datasets and interpreting the results. • Building dashboards using tools like Plotly or Dash.
7	• Implementing a complete machine learning pipeline from data collection to model deployment. • Working on real-world datasets and competitions (e.g., Kaggle). • Understanding the ethical implications and bias in machine learning.

7. Practical Works

S.N.	Practical works
1	Conduct an exploratory data analysis (EDA) on a public dataset.
2	Perform data manipulation tasks such as filtering, grouping, and summarizing.
3	Implement and compare different statistical techniques to analyze sample data (e.g., hypothesis testing, regression analysis).
4	Clean and preprocess a messy dataset (e.g., handling missing data, encoding categorical variables, feature scaling).
5	Implement different supervised learning algorithms (e.g., linear regression, decision trees) on a dataset.
6	Apply clustering techniques (e.g., K-means, hierarchical clustering) on a dataset and evaluate the clusters.
7	Perform a probabilistic model.
8	Apply anomaly detection methods in real world dataset.

8. Evaluation system and Students' Responsibilities

Evaluation System

In addition to the formal exam(s) conducted by the Office of the Controller of Examination of Pokhara University, the internal evaluation of a student may consist of class attendance, class participation, quizzes, assignments, presentations, written exams, etc. The tabular presentation of the evaluation system is as follows.

External Evaluation	Marks	Internal Evaluation	Marks
Semester-End Examination	50	Class attendance and participation	5
		Lab, Case study and Viva	15
		Internal Term Exam	30
Total External	50	Total Internal	50

Full Marks 50+50 = 100

Students' Responsibilities:
Each student must secure at least 45% marks in the internal evaluation with 80% attendance in the class to appear in the Semester End Examination. Failing to obtain such a score will be given NOT QUALIFIED (NQ) and the student will not be eligible to appear in the End-Term examinations. Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during the period. If a student fails to attend a formal exam, quiz, test, etc. there won't be any provision for a re-exam.

9. Prescribed Books and References

Text Book

Grus, J. *Data Science from Scratch: First Principles with Python*, Second Edition, O'Reilly Media.
Geron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, Second Edition, O'Reilly Media.
An Introduction to Statistical Learning by Gareth James et al.
O'Neil, C. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*, Crown Publishing Group.
Aggarwal, C. C. (2017). *Outlier analysis* (2nd ed.). Springer.

Reference Books

McKinney, W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*, Second Edition, O'Reilly Media.

Pokhara University
Faculty of Science and Technology

Course No.: CMP 340
Course title: **Software Design and Architecture (3-1-2)**
Nature of the course: Theory & Practical
Year: Second
Level: Bachelor

Full marks: 100
Pass marks: 45
Time per period: 1 hour
Total periods: 45
Program: BE

1. Course Description

Software Design and Architecture is an advanced course that builds on fundamental software engineering principles, focusing on designing robust, scalable, and maintainable software systems. Students will explore key concepts such as architectural styles, design patterns, and component-based design, learning how to apply these techniques to real-world projects. The course covers the decision-making process involved in selecting appropriate architectural solutions and navigating trade-offs between functional and non-functional requirements. Through case studies and hands-on projects, students will gain experience in designing software architectures that address complex system needs, ensuring performance, scalability, and security. This course prepares students for roles as software designers and architects in large-scale software development projects.

2. General Objectives

- To equip students with a deep understanding of architectural styles and design patterns to create scalable and maintainable software systems.
- To enable students to make informed architectural decisions by evaluating trade-offs and balancing system performance, security, and flexibility.
- To develop proficiency in applying advanced design principles such as SOLID, DRY, and component-based design for efficient system development.
- To provide practical knowledge of architecture validation methods to ensure that architectural designs meet both functional and non-functional requirements.
- To prepare students to design software for emerging technologies such as distributed systems, microservices, cloud-native applications, and highly available systems.

3. Methods of Instruction

3.1. General instructional Techniques: Lecture, discussion, readings.
3.2. Specific instructional Techniques: Lab works, case study

4. Contents in Detail.

Specific Objectives	Contents
- To understand the fundamental concepts of software design and architecture.	**Unit 1: Introduction to Software Design and Architecture (6 hrs)** 1.1 Overview of software design and architecture 1.2 Design levels: Architectural vs. detailed design 1.3 Design principles (modularity, abstraction, separation of concerns) 1.4 The role of the software architect 1.5 Key design goals: Performance, scalability, maintainability 1.6 Relationship between software design and software architecture 1.7 Object Oriented Software Development: Unified Software Development Process
- To explore various architectural styles and patterns and their impact on system quality. - To apply design patterns to solve common design challenges.	**Unit 2: Architectural Styles and Patterns (7 hrs)** 2.1 Common architectural styles (Layered, Client-server, Microservices, Event-driven, SOA) 2.2 Architectural patterns (MVC, Broker, Pipe and Filter) 2.3 Design patterns (Singleton, Factory, Observer, Strategy) 2.4 Choosing appropriate architectural styles and patterns based on system requirements 2.5 Impact of architecture on non-functional requirements (performance, scalability, security)
- To apply SOLID principles and best practices for designing modular, maintainable software. - To understand the importance of coupling, cohesion, and design for scalability.	**Unit 3: Software Design Principles and Best Practices (7 hrs)** 3.1 SOLID principles and their application 3.2 DRY (Don't Repeat Yourself) and KISS (Keep It Simple, Stupid) 3.3 Coupling and cohesion 3.4 Reusability, modularity, and maintainability in software design 3.5 Design for change and scalability

	3.6 Separation of concerns and information hiding 3.7 Case Study: Implementation of SOLID Principles in the Development.
- To design reusable and maintainable software components and manage component lifecycles. - To understand component composition, integration, and communication in large-scale systems.	**Unit 4: Component-Based Software Design (6 hrs)** 4.1 Introduction to component-based development 4.2 Components, interfaces, and contracts 4.3 Component composition, integration, and communication 4.4 Designing reusable and maintainable components 4.5 Component lifecycle management and versioning 4.6 Component-based design in large-scale systems
- To evaluate architectural trade-offs and balance functional and non-functional requirements.	**Unit 5: Architectural Decision-Making and Trade-offs (6 hrs)** 5.1 Architectural decision-making process 5.2 Balancing functional and non-functional requirements 5.3 Trade-offs in architecture (e.g., performance vs. scalability, security vs. flexibility) 5.4 Risk analysis and mitigation strategies in architecture 5.5 Justifying architectural decisions and documenting trade-offs 5.6 Case study: A system developed using Microservices Architecture

- To understand and apply methods for evaluating and validating software architectures.	**Unit 6: Software Architecture Evaluation and Validation (6 hrs)** 6.1 Architecture evaluation methods (ATAM, CBAM) 6.2 Scenario-based architecture validation 6.3 Architecture reviews and continuous validation 6.4 Assessing architecture against system qualities (performance, security, usability) 6.5 Tools for architecture evaluation and validation 6.6 Architecture evaluation in agile and DevOps environments
- To design for distributed systems, microservices, and cloud-native architecture. - To understand advanced design strategies for concurrency, fault tolerance, and high availability.	**Unit 7: Advanced Topics in Software Design and Architecture (7 hrs)** 7.1 Designing for distributed systems and cloud-native architecture 7.2 Microservices and serverless architecture 7.3 Domain-Driven Design (DDD) 7.4 Designing for concurrency and parallelism 7.5 Fault tolerance and high availability

5. List of Tutorials:

The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover all the required contents of this course.

S.N.	Tutorials
1	Case study on design Patterns Workshop
2	Case study on Domain-Driven Design (DDD)
3	Discussion on Microservices Architecture Simulation:
4	Case study on ATAM Evaluation
5	Discussion on Scenario-Based Architecture Evaluation:
6	Discussion on Clean Architecture Coding
7	Case study on Event-Driven Architecture Design

6.List of Practical

The following Practical works of 30 hours per group of maximum 24 students should be conducted to cover all the required contents of this course.

S.N.	Practical
1	Students will explore and apply common design patterns such as Singleton , Factory Observer, Strategy by working in pairs to implement a simple application..

7.

2	Students will model a given domain using Domain-Driven Design principles. They will identify key entities and relationships, create a domain model diagram, and explain their design choices to the class.
3	Students will design a microservices architecture for a specified use case, breaking down the system into services and defining their interactions. They will present their architecture and discuss deployment strategies and potential challenges.
4	Using the Architecture Tradeoff Analysis Method (ATAM), students will evaluate a provided software architecture based on quality attributes and trade-offs. They will complete an evaluation checklist and present their findings and recommendations.
5	Students will create and analyze scenarios to assess the effectiveness of a given software architecture. They will determine how well the architecture meets specific quality attributes and present their evaluation results.
6	Students will refactor a provided codebase to adhere to Clean Architecture principles, focusing on separation of concerns and maintainability. They will discuss their refactoring approach and the improvements made.
7	Students will design an event-driven system for a given problem, identifying events, event handlers, and communication patterns. They will present their system design and address potential challenges and scalability issues.

Evaluation system and Students' Responsibilities

Internal Evaluation

The internal evaluation of a student may consist of assignments, attendance, test-exams, term-exams, lab reports and projects etc. The tabular presentation of the internal evaluation is as follows:

External Evaluation	Marks	Internal Evaluation	Weight	Marks

Semester-End examination	50	Attendance	10%	50
		Tutorial/Case Studies	40%	
		Term exam	50%	
	50	Internal Final	100%	50
Full Marks 50+50= 100				

Student Responsibilities:

Each student must secure at least 45% marks in internal evaluation with 80% attendance in the class in order to appear in the Semester-End Examination. Failing to get such score will be given NOT QUALIFIED (NQ) and the student will not be eligible to appear the Semester-End Examination. Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during the period. If a student fails to attend a formal exam, test, etc. there won't be any provision for re-exam.

7. Prescribed Books and References

Books and Standards

1. Bass, L., Clements, P., & Kazman, R. (2021). *Software architecture in practice* (4th ed.). Addison-Wesley. ISBN: 978-0321815736
2. Evans, E. (2004). *Domain-driven design: Tackling complexity in the heart of software*. Addison-Wesley. ISBN: 978-0321125217
3. Kruchten, P. (2004). *The rational unified process: An introduction* (3rd ed.). Addison-Wesley. ISBN: 978-0321197700
4. IEEE Standards Association. (2017). *IEEE Std 1471-2000 - IEEE recommended practice for architectural description of software-intensive systems*.
5. ISO/IEC/IEEE 42010:2011 - Systems and software engineering — Architecture description.

Online Resources

1. Software Engineering Body of Knowledge (SWEBOK). (n.d.). *SWEBOK Guide*.
2. IEEE Xplore Digital Library. *IEEE software engineering standards*.
3. Kazman, R., Bass, L., & Abowd, G. (2000). Scenario-based analysis of software architecture. *ACM SIGSOFT Software Engineering Notes, 25*(1), 37-48.

Supplement Materials

Books

1. Kleppmann, M. (2017). *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. O'Reilly Media. ISBN: 978-1449373320
2. Martin, R. C. (2017). *Clean architecture: A craftsman's guide to software structure and design*. Prentice Hall. ISBN: 978-0134494166